AFIT/GA/ENY/91D-7

PERTURBATION THEORY FOR RESTRICTED THREE-BODY ORBITS

THESIS

David A. Ross, Captain, USAF

AFIT/GA/ENY/91D-7

Approved for public release; distribution unlimited

AFIT/GA/ENY/91D-7

PERTURBATION THEORY FOR RESTRICTED THREE-BODY ORBITS

THESIS

Presented to the Faculty of the School of Engineering

of the Air Force Institute of Technology

Air University

In Partial Fulfillment of the

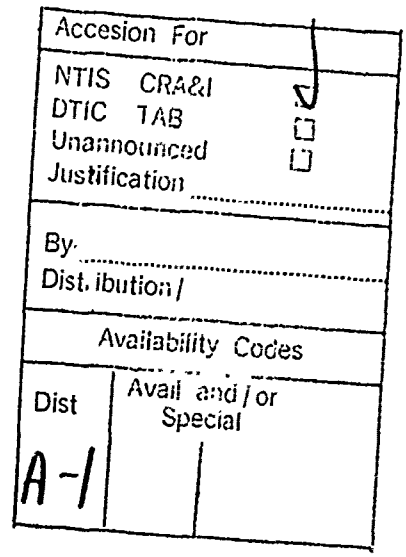
Requirements for the Degree of

Master of Science in Astronautical Engineering

David A. Ross, B.S.

Captain, USAF

December 1991



Preface

The purpose of this study was to validate the three-body perturbation theory described by Dr. William Wiesel, in his paper entitled, "Perturbation Theory in the Vicinity of a Periodic Orbit by Repeated Linear Transformations". Mastering these techniques would be a necessary first step to navigating a highly perturbed region of space, such as that surrounding the Martian moon Phobos. Any further developments in the understanding of time-periodic systems in general, could reap great rewards in several non-astronautical areas as well. The dynamics of helicopter blades is a good example.

This entire study would not have been possible if not for the extensive help I received from Dr. Wiesel himself. The process began with many hours of tutoring on the major technical points of the thesis. Each area discussed eventually became one of the six different programs used during the course of the study. I was given free reign of several different programs, subroutines, and hard to find text books. If Dr. Wiesel didn't already have a similar program to be modified, he would help me lay out an algorithm that I could later encode. Dr. Wiesel also devised several analytical tests that could be made to ensure program accuracy.

For all of the reasons stated above, as well as the thrill of showing someone that their ideas actually work, I consider this thesis an overwhelming success. Yet there are still a great deal of unanswered questions, and program modifications to make.

I would also like to take this opportunity to thank my wife Beth, for her support and understanding during this last year and a half. Without her, the answer to the question, why bother ?, would not have been so apparent.

Finally, I hereby dedicate this entire work to my very special other Mom, Lynne, whose passing has opened my eyes to what is truly important in this world.

David A. Ross

Table of Contents

List of Figures

List of Symbols

α_i Expanded Hamiltonian Coefficient

δ Infinitesimal Variable

Λ_{ij} Element of Eigenvector Matrix F

μ Mass / Distance Parameter

T Orbital Period

τ Dimensionless Orbital Period

Φ State Transition Matrix

ϕ_{ij} Element of State Transition Matrix

χ State Vector

$\chi 0$ State Vector of a Periodic Orbit

$\delta\chi$ State Vector for a Nearly-Periodic Orbit

Ω Portion of Jacobi's Integral

ω Imaginary Part of Non-Zero Poincaré Exponent

A Linearization of the Dynamics

b Modal State Vector

C Jacobi Constant, Interchangeable with Hamiltonian Constant

com System Center of Gravity

F Eigenvector Matrix

f() Generic Functional Relationship

F_2 Generating Function for Canonical Transformation

G Gravitational Constant

$g_1()$ Generic Functional Relationship

$g_2()$ Generic Functional Relationship

H Hamiltonian Constant

J Poincaré Exponent Matrix, Diagonal Block Entries

K	New Hamiltonian
L	Lagrangian
M	Poincaré Exponent Matrix, Jordan Normal Form
M_i	Planetary Masses
m_i	Dimensionless Planetary Masses
n	Number of Masses in a System, Mean Motion of the Primaries
p_i	Momenta Conjugate to Coordinate Variables q_i
$p0_i$	Periodic Momenta Conjugate to Coordinates $q0_i$
δp_i	Nearly-Periodic Momenta Conjugate to Coordinates δq_i
q_i	Coordinate Variables for Restricted Three-Body System
$q0_i$	Periodic Coordinates
δq_i	Nearly-Periodic Coordinates
r	Position Vector for Massless Third Body
r_1	Position Vector for Primary m_1
r_2	Position Vector for Primary m_2
S_i	Magnitude of Orbital Radii
s_i	Dimensionless Magnitude of Orbital Radii
T	System Kinetic Energy
V	System Potential Energy
v	Velocity Vector for Third Body
w	Angular Velocity Vector For Rotating Frame
x	Coordinate Variable in SOS Coordinate Frame
y	Coordinate Variable in SOS Coordinate Frame
$x0$	Initial Condition
$y0$	Initial Condition
Z	Correlation Matrix

ABSTRACT

A perturbation theory for restricted three-body orbits, using a periodic trajectory as a reference solution, is investigated. The nearly-periodic equations of motion are derived by analogy to a linearization about an equilibrium point. In this case, the linearization produces a set of time-periodic equations of motion that, according to Floquet, are completely solved by a periodic trajectory.

The four-dimensional phase space of the restricted three-body problem is first surveyed for regions of periodic motion, via the surface of section phase plot. Upon extraction of a periodic orbit, nearly-periodic orbits are integrated. The integrated state vector is routinely sampled, and then twice transformed into a new set of variables. The first translates the frame center to the periodic trajectory. The second, or modal transformation, projects the coordinates along their eigenvectors. The transformations are highly useful, since two of four new variables are constant within a finite region surrounding the periodic reference. Plots of the two variables are offered as an exact representation of a nearly-periodic trajectory, while plots of the constants over time, trace the boundaries of the nearly-periodic region.

After the original Hamiltonian is canonically transformed into the new variables, it is expanded in a Taylor's series. Several of the terms are either simplified or annihilated completely. The expansion is then truncated after four terms,

leaving a readily differentiable expression from which to derive the nearly-periodic equations of motion. The expanded trajectories are then compared to the exact ones, over a wide range of values.

As was expected, a significant region exists where the expanded equations of motion accurately reproduce the concentric circular paths shown to exist by the transformed case. As the initial displacements from the periodic trajectory are increased, the expanded trajectories fail to accurately model the transients observed in the exact case. These exact case, orbital irregularities occur because the displacements from the periodic orbit are no longer small enough to be represented by a linearization of the dynamics. The expanded trajectories fail to recreate this non-linear behavior because most of the Hamiltonian terms responsible have been truncated. Therefore, before the complete perturbation solution can be constructed, the expanded, nearly-periodic equations of motion should be derived again using more than four terms of the expanded Hamiltonian.

PERTURBATION THEORY FOR RESTRICTED THREE-BODY ORBITS

I. Introduction

Before the astrodynamics of man-made objects in space can be fully understood, one must first comprehend basic planetary motion. Thanks to Sir Isaac Newton and his three laws of motion, and to Johann Kepler for his three laws of orbital motion, it can be shown that nearly all astrodynamical systems are dominated by a single conservative force known as gravity. In fact, the most general description of the motion of a collection of objects in space is defined by the n-body problem.

In an n-body system, the n^{th} body is acted upon by the other n-1 gravitational masses present. In this way, the motion of any mass in a system affects and is affected by every other mass in the system. The overwhelming task of representing each body is well illustrated by Wiesel.

> Our own solar system consists of one star, nine planets, over fifty moons, tens of thousands of asteroids, and millions of comets. The description of the motion of this system is clearly important, but an exact solution to this problem has not been found in over three hundred years of study. (8:33)

Therefore, the use of the exact n-body description of a dynamical system is not simply a nuisance, it is virtually impossible to implement.

The simplest and most drastic approximation to the n-body problem is known appropriately as the two-body problem. Here,

only two point masses are considered to exist within the dynamical system. The primary masses are then constrained their by mutual gravitational attraction. Specifically, both bodies remain in a circular orbit about the system center of mass point. This particular arrangement is very special in the field of astrodynamics since, "it is the only gravitational problem for which a closed-form solution has been found" (8:45). Additionally, one might suspect the accuracy of such a severely truncated formulation. Surprisingly enough, "most systems encountered in orbital mechanics are nearly perfect two-body problems, with only small perturbations from two-body motion" (9:75).

There are situations, unfortunately, where the small perturbation assumption of two-body perturbation theory is violated. Such a case occurs in the vicinity of the Martian moon Phobos. The orbit about Phobos is so dramatically effected by the gravitational pull from Mars, that a simple two-body approximation can't simulate the true dynamics of the region. According to Szebehely,

> Entry into celestial mechanics and space dynamics can be gained by the study of the problem of two bodies. To penetrate the fundamental problems, the number of participating bodies must be increased from two to three. (5:v)

Clearly, in highly perturbed dynamical systems, higher orders of approximation must replace the tractable two-body scenario.

II. Historical Development

This study is the first of it's kind. An extensive literature search of four data bases, the AFIT library, and a tedious bout with the Science Citation Index, produced nothing. The only sources of information available on the development of a restricted three-body perturbation theory using a periodic orbit as the reference solution, were Dr. Wiesel's paper on the subject and Dr. Wiesel himself. As described in his previous work, Dr. Wiesel contends that,

> It is common for researchers working with periodic orbits, to also solve the associated Floquet problem in order to derive stability information on the orbit. It is far less common to make use of eigenvectors of the linearized system, or to use a periodic orbit as a reference solution for perturbation theory. (7:231)

While it appears that this study is similar to others in it's content, it is quite original in it's purpose. It is unique to use the eigenvectors and Poincaré exponents of the periodic trajectory, to canonically transform the generic equations of motion into nearly-periodic ones. By analogy to classical two-body perturbation theory the periodic trajectory serves as the reference solution, while displacements from this reference are treated as small perturbations.

III. Theory

The Restricted Three-Body Problem

The complete three-body formulation at first glance, might not appear to be much more difficult to solve than the two-body problem. While the inclusion of another gravitational mass into a two-body system would further complicate the dynamics, one would still hope to find at least a partial solution. In reality, however, the problem of three-bodies is completely unsolvable without imposing the restrictions first defined by Leonard Euler in 1772.

In the restricted three-body problem, it is assumed that two of the bodies are tremendously more massive than the third. The motion of the third body is then governed by the gravitational pull of the two primary bodies. Conversely, the motion of the primaries is unaffected by the third body, and is completely described by the two-body solution. In this way, the restricted problem can be considered a one-body problem, because only the equations of motion of the third body are of interest.

Before deriving the equations of motion, non-dimensional variable definitions for mass, length, and time must be introduced. First, the non-dimensional masses m_1, m_2, and m_3 are defined by

$$m_1 = \frac{M_1}{M_1+M_2} \quad m_2 = \frac{M_2}{M_1+M_2} \quad m_3 = \frac{M_3}{M_1+M_2} \approx 0 \tag{1}$$

where M_1 and M_2 are the masses of the primary bodies, and M_3 is

the mass of the third body (see figure 1). Second, the non-dimensional radii, s_1, s_2, connecting the primary masses to the center of mass are given by

$$s_1 = \frac{S_1}{S_1+S_2} \qquad s_2 = \frac{S_2}{S_1+S_2} \tag{2}$$

where S_1 and S_2 are the actual radius magnitudes. The center of mass position s_1 as measured from m_1, is then calculated by

$$s_1 = \frac{\sum m_i s_i}{\sum m_i} = \frac{m_1 \times 0 + m_2 \times (s_1+s_2)}{m_1+m_2} = m_2 \tag{3}$$

Remembering that

$$s_1+s_2 = m_1+m_2 = 1 \tag{4}$$

then these parameters may be redefined by a single parameter μ.

$$s_1 = m_2 = \mu \qquad s_2 = m_1 = 1-\mu \tag{5}$$

Third, the non-dimensional orbit period τ is defined as

$$\tau = T\left[\frac{G(M_1+M_2)}{(S_1+S_2)^3}\right]^{\frac{1}{2}} = 2\pi\left[\frac{(S_1+S_2)^3}{G(M_1+M_2)}\right]^{\frac{1}{2}}\left[\frac{G(M_1+M_2)}{(S_1+S_2)^3}\right]^{\frac{1}{2}} = 2\pi \tag{6}$$

where G is the universal gravitation constant. As a consequence of eq(6), the non-dimensional angular velocity of the rotating coordinate frame, w, may be derived from the mean motion of the orbiting primaries.

$$w = n\left[\frac{(S_1+S_2)^3}{G(M_1+M_2)}\right]^{\frac{1}{2}} = \left[\frac{G(M_1+M_2)}{(S_1+S_2)^3}\right]^{\frac{1}{2}}\left[\frac{(S_1+S_2)^3}{G(M_1+M_2)}\right]^{\frac{1}{2}} = 1 \tag{7}$$

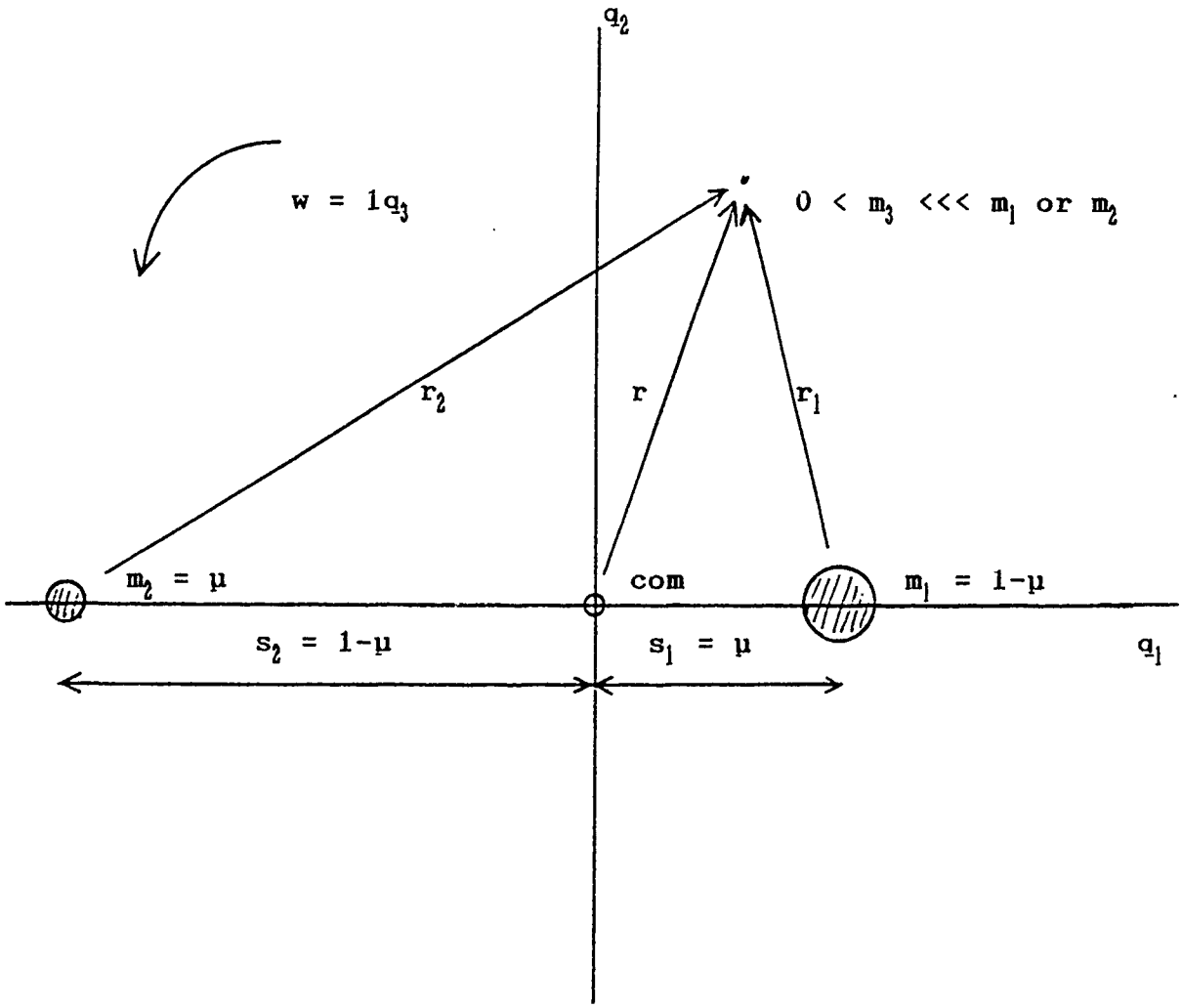


Figure 1. Reference Frame for the Restricted Three-Body System

The Restricted Three-Body Equations of Motion

In his treatise on the problem of three bodies, Victor Szebehely devotes the entire first chapter to the description and derivation of the equations of motion for the restricted three-body problem (5:7-22). The equations of motion to be presented differ from those of Szebehely, since his final result is a set of two second-order differential equations (3:22). Here, four first-order differential equations are derived, which are dynamically equivalent to those of Szebehely.

The position vector for the third-body, and the angular velocity vector for the rotating coordinate frame can be directly observed from figure 1.

$$\bar{r} = q_1\hat{q}_1 + q_2\hat{q}_2 \tag{8}$$

$$\bar{w} = 1\hat{q}_3 \tag{9}$$

where

$$\hat{q}_1 , \hat{q}_2 , \hat{q}_3 \tag{10}$$

form a set of orthogonal unit vectors. The velocity vector is

$$\frac{d}{dt}\bar{r} = \bar{v} = (\dot{q}_1 - q_2)\hat{q}_1 + (\dot{q}_2 + q_1)\hat{q}_2 \tag{11}$$

and the dimensionless kinetic energy of the third-body is then

$$T = \tfrac{1}{2}m[\bar{v}\cdot\bar{v}] = \tfrac{1}{2}m_3\left[(\dot{q}_1 - q_2)^2 + (\dot{q}_2 + q_1)^2\right] \tag{12}$$

The non-dimensional potential energy of m_3 is purely gravitational, and is given by

$$V = \left[\frac{V'}{G(M_1+M_2)}\right]\left[\frac{-GM_1M_3}{r_1} + \frac{-GM_2M_3}{r_2}\right] = \frac{-m_3(1-\mu)}{r_1} + \frac{-m_3(\mu)}{r_2} \qquad (13)$$

where

$$\begin{aligned} r_1 &= \left[(q_1-\mu)^2+(q_2)^2\right]^{\frac{1}{2}} \\ r_2 &= \left[(q_1+1-\mu)^2+(q_2)^2\right]^{\frac{1}{2}} \end{aligned} \qquad (14)$$

the dimensionless Lagrangian is then

$$L = \frac{L'}{m_3} = \tfrac{1}{2}\left[(\dot{q}_1-q_2)^2+(\dot{q}_2+q_1)^2\right] + \frac{(1-\mu)}{r_1} + \frac{(\mu)}{r_2} \qquad (15)$$

The conjugate momentum terms, p_1 and p_2, needed to complete the state vector

$$\bar{\chi}(t) = \begin{bmatrix} q_1(t) \\ p_1(t) \\ q_2(t) \\ p_2(t) \end{bmatrix} \qquad (16)$$

are constructed by differentiating the Lagrangian.

$$p_1 = \frac{\partial L}{\partial \dot{q}_1} = \dot{q}_1-q_2 \qquad p_2 = \frac{\partial L}{\partial \dot{q}_2} = \dot{q}_2+q_1 \qquad (17)$$

A rearrangement and substitution of the momenta into the Lagrangian then yields

$$L = \tfrac{1}{2}\left(p_1^2+p_2^2\right) + \frac{1-\mu}{r_1} + \frac{\mu}{r_2} \qquad (18)$$

Thus, the system Hamiltonian is

$$H = \Sigma p_i\dot{q}_i - L = \tfrac{1}{2}\left(p_1^2+p_2^2\right) + p_1q_2 - p_2q_1 - \frac{1-\mu}{r_1} - \frac{\mu}{r_2} \qquad (19)$$

The equations of motion for the restricted third-body may then be obtained via Hamilton's equations.

$$\begin{aligned}
\dot{q}_1 &= \frac{\partial H}{\partial p_1} = p_1 + q_2 \\
\dot{p}_1 &= -\frac{\partial H}{\partial q_1} = p_2 - \frac{(1-\mu)(q_1+\mu)}{r_1^3} - \frac{\mu(q_1+1-\mu)}{r_2^3} \\
\dot{q}_2 &= \frac{\partial H}{\partial p_2} = p_2 - q_1 \\
\dot{p}_2 &= -\frac{\partial H}{\partial q_2} = -p_1 - \frac{(1-\mu)q_2}{r_1^3} - \frac{\mu q_2}{r_2^3}
\end{aligned} \tag{20}$$

Periodic Orbits and the Equations of Variation

Periodic orbits are a very special sub-set of restricted three-body orbits. In general, a periodic orbit is nothing more than an orbit that closes upon itself after each revolution, and nothing less than one of the few known solutions to restricted three-body dynamics. Since it is the point of this study to gain insight into nearly periodic orbits, an understanding of the periodic case must first be obtained.

A periodic orbit will always return to it's original state after each integer multiple of it's period. Because of this, a periodic orbit may be calculated by iteratively narrowing the difference between the initial and final state conditions. In general, periodicity is obtained when

$$\bar{\chi}(0) = \bar{\chi}(\tau) \tag{21}$$

In practice, once a set of initial conditions have been chosen and the orbit integrated, one will find that the initial and final conditions will not agree. Therefore, before the next iteration, the initial conditions must be adjusted based on the error found in the final conditions of the last integration. This can only be accomplished if dynamical information about nearby trajectories is made available. This information is contained within the equations of variation.

In vector form, the equations of motion may be rewritten as

$$\frac{d}{dt}\bar{\chi} = \bar{f}(\bar{\chi}) \tag{22}$$

Similarly, if we define the state of a nearby trajectory as the vector sum of the current trajectory and the displacement vector separating the two trajectories, then

$$\bar{\chi}_{nearby} = \bar{\chi} + \delta\bar{\chi} \tag{23}$$

Substituting this result into the equations of motion, we get

$$\frac{d}{dt}\bar{\chi} + \frac{d}{dt}(\delta\bar{\chi}) = \bar{f}(\bar{\chi}+\delta\bar{\chi}) \tag{24}$$

After expansion in a Taylor's series, centered about the original trajectory, the equations of motion become

$$\frac{d}{dt}\bar{\chi} + \frac{d}{dt}(\delta\bar{\chi}) = \bar{f}(\bar{\chi}) + \left.\frac{\partial \bar{f}}{\partial \bar{\chi}}\right|_{\bar{\chi}} \delta\bar{\chi} + O|\delta\bar{\chi}|^2 \tag{25}$$

If we assume a first-order expansion and then subtract the original equations of motion, we obtain the equations of variation.

$$\frac{d}{dt}(\delta\bar{\chi}) = \left.\frac{\partial \bar{f}}{\partial \bar{\chi}}\right|_{\bar{x}} \delta\bar{\chi} \tag{26}$$

In vector form, Hamilton's equations may be written as

$$\bar{f}(\bar{\chi}) = Z\frac{\partial H}{\partial \chi} \tag{27}$$

where

$$Z = \begin{bmatrix} 0 & 1 & 0 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & -1 & 0 \end{bmatrix} \tag{28}$$

Differentiating Hamilton's equations with respect to the state vector yields

$$\left.\frac{\partial \bar{f}}{\partial \bar{\chi}}\right|_{\bar{x}} = Z\,\frac{\partial^2 H}{\partial \bar{\chi}^2} = A(t) \tag{29}$$

where $A(t)$ is known as the linearization of the dynamics. Thus, the equations of variation become

$$\frac{d}{dt}(\delta\bar{\chi}) = A(t)\,\delta\bar{\chi} \tag{30}$$

expanding $A(t)$,

$$A(t) = \begin{bmatrix} 0 & 1 & 1 & 0 \\ -H_{11} & 0 & -H_{13} & 1 \\ -1 & 0 & 0 & 1 \\ -H_{31} & -1 & -H_{33} & 0 \end{bmatrix} \tag{31}$$

where

$$H_{11} = \frac{-3(q_1-\mu)^2(1-\mu)}{r_1^5} - \frac{3(q_1+1-\mu)^2\mu}{r_2^5} + \frac{1-\mu}{r_1^3} + \frac{\mu}{r_2^3}$$

$$H_{13} = \frac{-3(q_1-\mu)q_2(1-\mu)}{r_1^5} - \frac{3(q_1+1-\mu)q_2\mu}{r_2^5} \qquad (32)$$

$$H_{31} = H_{13}$$

$$H_{33} = \frac{-3q_2^2(1-\mu)}{r_1^5} - \frac{3q_2^2\mu}{r_2^5} + \frac{1-\mu}{r_1^3} + \frac{\mu}{r_2^3}$$

Since the equations of motion form a set of first-order, time-varying, linear, differential equations, the general solution may be constructed from the fundamental set of solutions (6:61).

$$\delta\bar{\chi}(t) = \Phi(t)\ \delta\bar{\chi}(0) \qquad (33)$$

where Φ is the solution to the differential equation

$$\frac{d}{dt}\Phi(t) = A(t)\ \Phi(t) \qquad (34)$$

which in turn must be integrated along with the equations of motion.

Upon completion of an integration, the solution to the equations of variation may be constructed as

$$\begin{bmatrix} \delta q_1(\tau) \\ \delta p_1(\tau) \\ \delta q_2(\tau) \\ \delta p_2(\tau) \end{bmatrix} = \begin{bmatrix} \phi_{11} & \phi_{12} & \phi_{13} & \phi_{14} \\ \phi_{21} & \phi_{22} & \phi_{23} & \phi_{24} \\ \phi_{31} & \phi_{32} & \phi_{33} & \phi_{34} \\ \phi_{41} & \phi_{42} & \phi_{43} & \phi_{44} \end{bmatrix} \begin{bmatrix} \delta q_1(0) \\ \delta p_1(0) \\ \delta q_2(0) \\ \delta p_2(0) \end{bmatrix} \qquad (35)$$

If in the selection of the initial conditions, we restrict $\delta p_1(0)$ and $\delta q_2(0)$ to zero, then the orbit must not only close upon itself, but must do so intersecting the q_1 axis perpendicularly.

The benefits of this restriction are three fold. First, the number of initial conditions that must be correctly determined is halved. Second, the final values of δp_1 and δq_2 are the actual error in the boundary conditions. Third, only the portion of eq(35) that relates $\delta p_1(\tau)$ and $\delta q_2(\tau)$ to $\delta q_1(0)$ and $\delta p_2(0)$ is relevant. Therefore, eq(35) may be simplified to

$$\begin{bmatrix} \delta p_1(\tau) \\ \delta q_2(\tau) \end{bmatrix} = \begin{bmatrix} \Phi_{21} & \Phi_{24} \\ \Phi_{31} & \Phi_{34} \end{bmatrix} \begin{bmatrix} \delta q_1(0) \\ \delta p_2(0) \end{bmatrix} \tag{36}$$

and after matrix inversion to

$$\begin{bmatrix} \delta q_1(0) \\ \delta p_2(0) \end{bmatrix} = \begin{bmatrix} \Phi_{21} & \Phi_{24} \\ \Phi_{31} & \Phi_{34} \end{bmatrix}^{-1} \begin{bmatrix} \delta p_1(\tau) \\ \delta q_2(\tau) \end{bmatrix} \tag{37}$$

which yields the necessary relationship between initial and final conditions to find a periodic orbit.

Initial Condition Determination

Since the equations of variation are a linearized approximation of the true system dynamics, their use in an iteration scheme is limited. More to the point, the iteration scheme will only converge if the initial conditions chosen produce a nearly-periodic orbit. Thus, a solution may be extracted only if it has already been approximated.

The restricted three-body problem is spanned by four dimensions, and is solvable only by four exact integrals of motion. It was proven by Henri Poincaré, however, that

> The Hamiltonian is the only analytic integral of the motion. If other so called 'quasi integrals' exist,

> they are not analytic functions of the system coordinates, momenta, and time. (9:132)

Therefore, the periodic solution we seek must involve the Hamiltonian, and three of these 'quasi integrals'.

Given the absence of an analytical approach, it appears that periodic regions may only be identified through numerical search. Fortunately, such methods have been well developed and widely used, given the availability of fast and powerful computers. William Jefferys compiled an extensive catalog of restricted three-body phase plots, known as surface of section plots (3:1). These plots allow the user to graphically locate regions of periodicity, and to identify sufficiently periodic sets of initial conditions.

The Surface of Section

The equations of motion as derived by Jefferys differ from Szebehely's formulation in two ways. First, the coordinate frame is not centered at the system center of mass point, but has been translated a distance μ to the center of the primary body of mass $1-\mu$. (see figure 2) Second, the Jacobi constant C, is used instead of the system Hamiltonian, H. Therefore, in order to obtain Jefferys' results using the dynamics already presented, two transformation relationships must be established.

The relationship between coordinates is simply

$$\begin{aligned} q_1 &= x+\mu \\ q_2 &= y \end{aligned} \tag{38}$$

while the relationship between constants of motion is a bit more

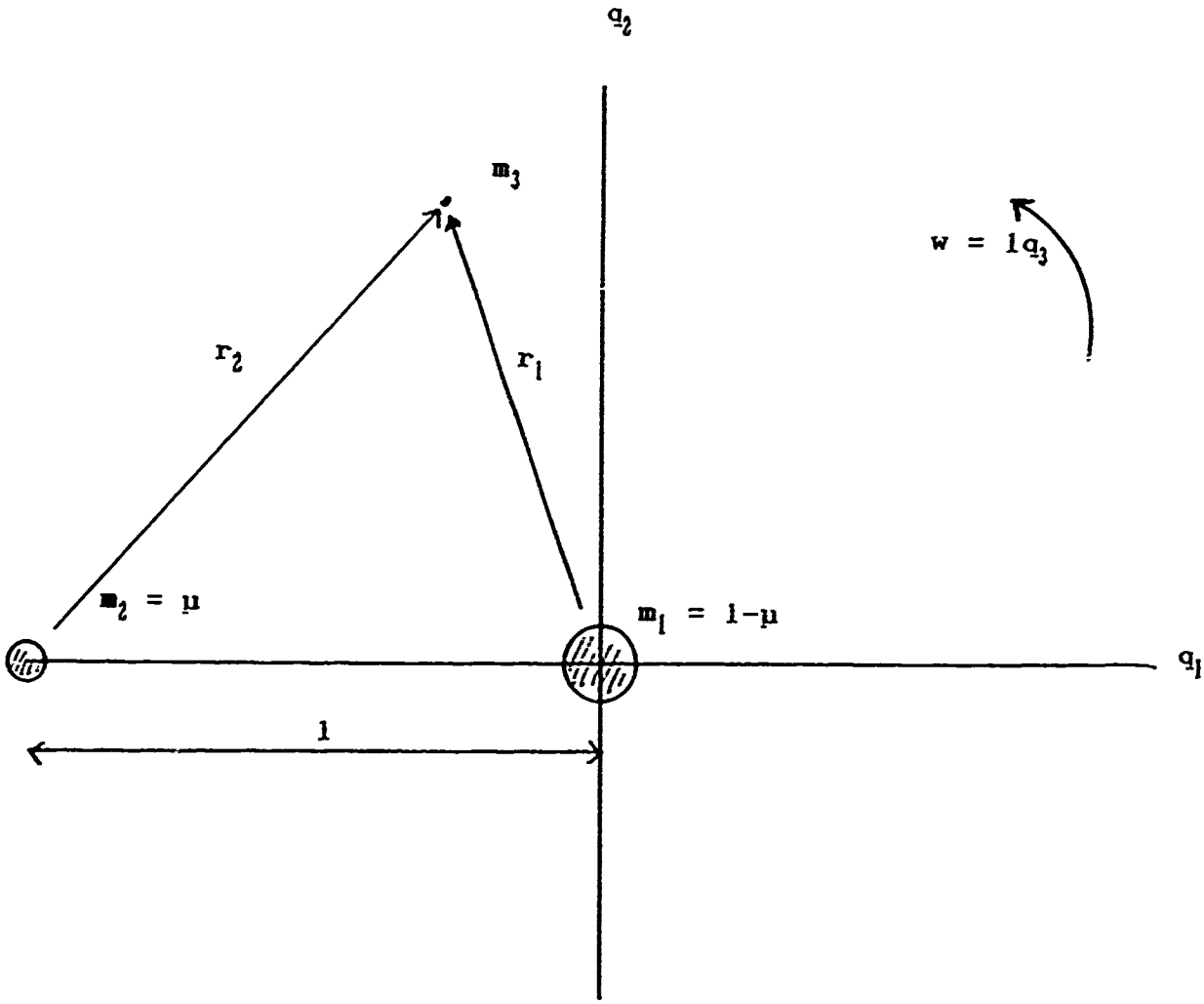


Figure 2. Reference Frame Used in the Surface of Section

involved. Jefferys uses the Jacobi integral in the form

$$\dot{x}^2 + \dot{y}^2 = 2\Omega - C \tag{39}$$

where

$$\Omega = (1-\mu)\left[\frac{1}{r_1} + \frac{r_1^2}{2}\right] + \mu\left[\frac{1}{r_2} + \frac{r_2^2}{2}\right]$$

$$r_1 = [x^2+y^2]^{\frac{1}{2}} \qquad r_2 = [(x+1)^2+y^2]^{\frac{1}{2}}$$

from Szebehely's equations of motion

$$\dot{q}_1 = p_1+q_2 \qquad \dot{q}_2 = p_2-q_1 \tag{41}$$

so after substitution, Jacobi's integral becomes

$$(p_1+q_2)^2 + (p_2-q_1)^2 = 2\left[\frac{1-\mu}{r_1} + \frac{\mu}{r_2} + \frac{(1-\mu)r_1^2}{2} + \frac{\mu r_2^2}{2}\right] - C \tag{42}$$

grouping the terms found in the Hamiltonian to the left

$$\begin{aligned} H &= \tfrac{1}{2}(p_1^2+p_2^2) + p_1q_2 - p_2q_1 - \frac{\mu}{r_2} - \frac{1-\mu}{r_1} \\ &= \tfrac{1}{2}\left[-(q_1^2+q_2^2) + (1-\mu)r_1^2 + \mu r_2^2 - C\right] \end{aligned} \tag{43}$$

then by substituting the radius terms

$$2H = -(q_1^2+q_2^2) + (1-\mu)\left[(q_1-\mu)^2+q_2^2\right] + \mu\left[(q_1-\mu+1)^2+q_2^2\right] - C \tag{44}$$

which simplifies nicely to

$$2H = \mu(1-\mu) - C \tag{45}$$

Thus, the transformation between the derivations of Szebehely and Jefferys is complete.

The surface of section does not plot orbit trajectories per se. Rather, it illustrates the behavior of all possible trajectories in a particular finite portion of the phase space.

> Because the Hamiltonian is a constant of the motion, the third body m_3 is constrained to move on a three dimensional manifold embedded in the four dimensional phase space. If another independent integral exists for this orbit, then the third body would then be constrained to move on a two-dimensional manifold embedded in the three-dimensional phase space. (3:6)

The reduction of dimension from four to three, occurs because specification of the Hamiltonian constant H and any three state variables, dictates the value of the fourth. In this way, the state vector may be transformed to include three state variables and one constant.

$$\bar{\chi}(t) = \begin{bmatrix} q_1 \\ p_1 \\ q_2 \\ p_2 \end{bmatrix} = \begin{bmatrix} q_1 \\ p_1 \\ q_2 \\ f(q_1, p_1, q_2, H, \mu) \end{bmatrix} \tag{46}$$

The dimension of the problem is further simplified by arbitrarily specifying a plane, through which all of the two-dimensional trajectories of interest must pass. The plane chosen by Jefferys was

$$x\dot{x} + y\dot{y} = q_1 p_1 + q_2 p_2 - \mu(p_1 + q_2) = 0 \tag{47}$$

since, "it represents the condition for an orbit to have a pericenter or apocenter (i.e., the point nearest to or farthest from the primary $1-\mu$)" (3:8). Again, by algebraic manipulation, the state vector may now be written as

$$\bar{\chi}(t) = \begin{bmatrix} q_1 \\ g_1(q_1, q_2, H, \mu) \\ q_2 \\ g_2(q_1, q_2, H, \mu) \end{bmatrix} \tag{48}$$

Therefore, by specifying a value for the Hamiltonian, the arbitrary plane, and the parameter μ, the dimension of the problem is reduced from four to two. In order to find periodic regions in the remaining two-space, Jefferys contends

> If a second integral exists, the intersection of the two-dimensional manifold on which the particle is constrained to move with the arbitrary plane will be one-dimensional, in general (i.e., a set of closed curves). On the other hand, if no such integral exists, then the intersection will not be restricted to one-dimensional sets in the arbitrary plane. (3:6)

Thus, we now have a way to identify periodic regions in the phase space, regions where two integrals of motion exist.

Figure 3 represents a precessing elliptical trajectory about primary $1-\mu$. By comparison, figure 4, shows which points from this trajectory actually intersect the plane of interest. In this case, the points of apogee and perigee constitute the only points on the rotating ellipse that pass through the plane. Therefore, the surface of section appears as two circles that trace the path of the apogee and perigee points, as the elliptical orbit precesses.

In general, on most surface of section plots, the location of the primaries and mass m_3 are omitted. Therefore, one should remember that the plot is always centered about the primary mass $1-\mu$, with the other primary mass, μ, located at the point $(-1,0)$. The actual location of mass m_3 is of little use and is never

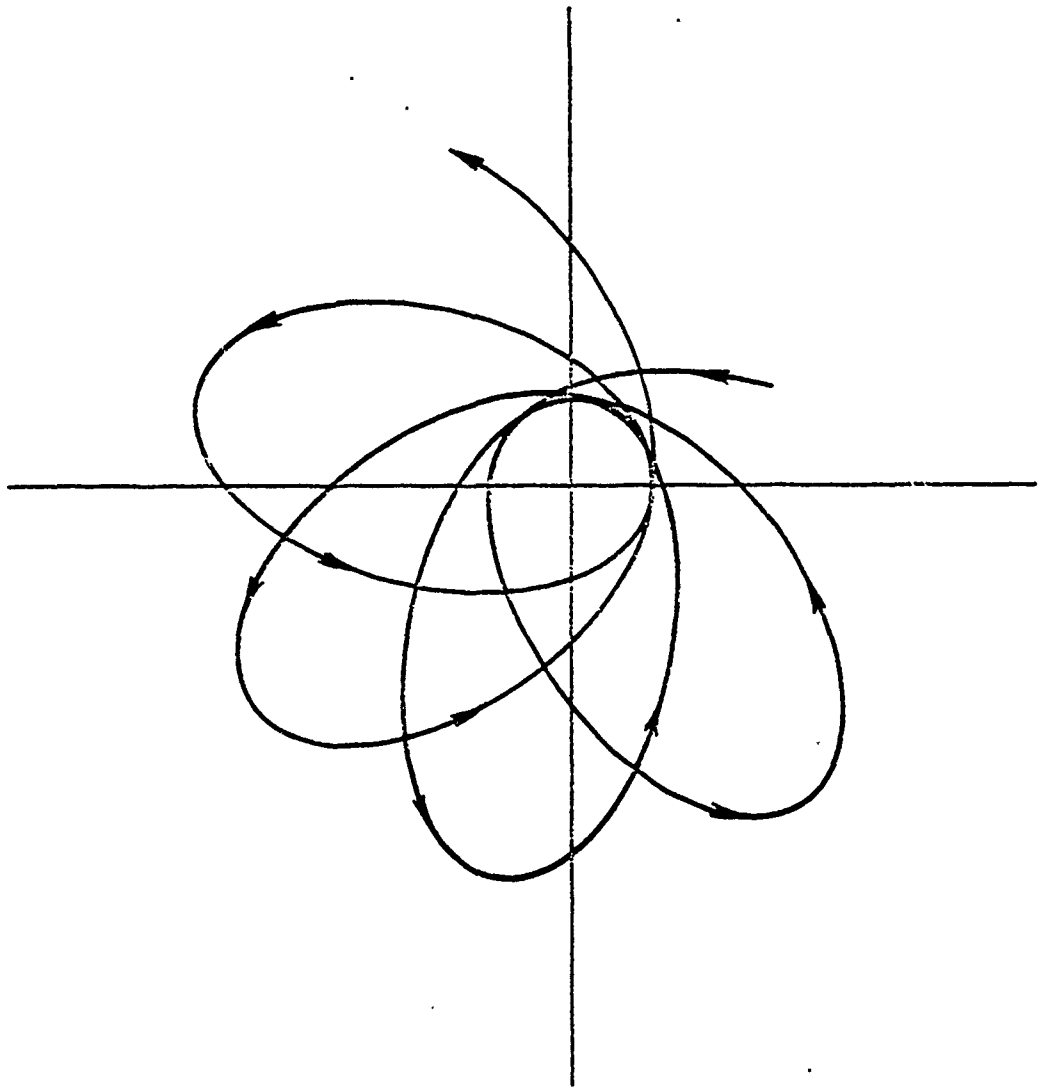


Figure 3. An Elliptical Trajectory Precessing About Primary 1-μ

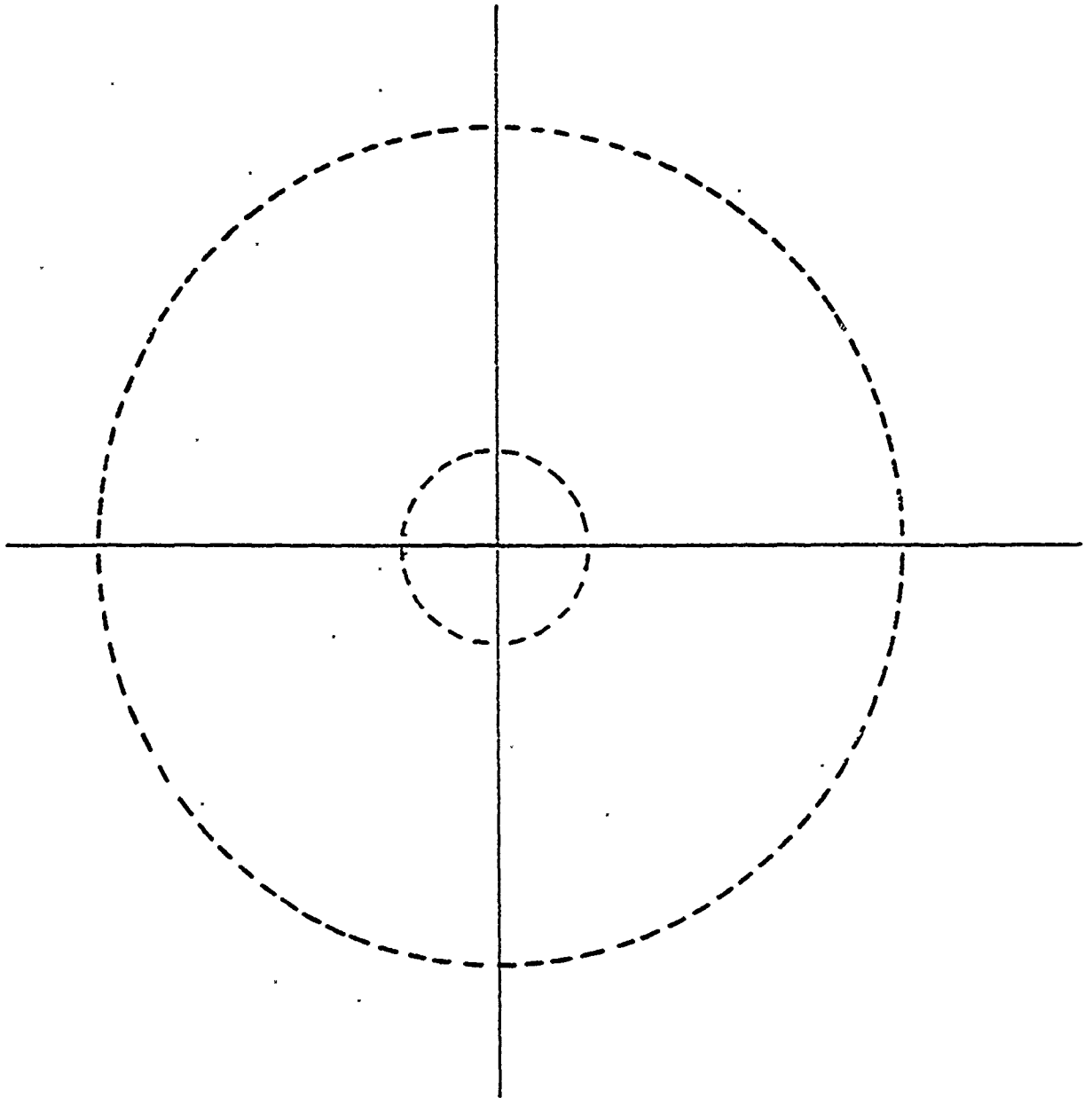


Figure 4. The Surface of Section of the Elliptical Trajectory

recorded. All that is really important here are the size, shape, and location of closed curves on the arbitrary plane.

Since it is the purpose of these plots to describe the dynamics of a region, several trajectories must be integrated and overlaid to produce a meaningful surface of section plot. Figure 5 is a surface of section plot for the sun-Jupiter system. The value of μ is small, which allows m_3 to orbit the primary $1-\mu$ in gently perturbed two-body fashion. Figure 6, on the other hand, is a good example of the highly perturbed, non-two-body case. Here, both primaries are of a size and proximity, that they grapple continuously for dynamical control of m_3. On either plot, the concentric enclosed island structures indicate regions of periodic motion. These regions are all centered by a single point that is a periodic solution.

The Nearly-Periodic Trajectory in Modal Variables

To summarize, the existence of Jacobi's integral reduces the dimension of the phase space from four to three. This is true anywhere in the restricted three-body phase space. Locally, however, a closed curve on the surface of section plot implies the presence of a second integral. A second integral further confines the dynamics to a two-dimensional manifold, embedded within the four-dimensional phase space. The true dynamical nature of these local two-dimensional manifolds is distorted on the surface of section plot, because it is a projection of the two-dimensional manifold onto the arbitrarily defined plane.

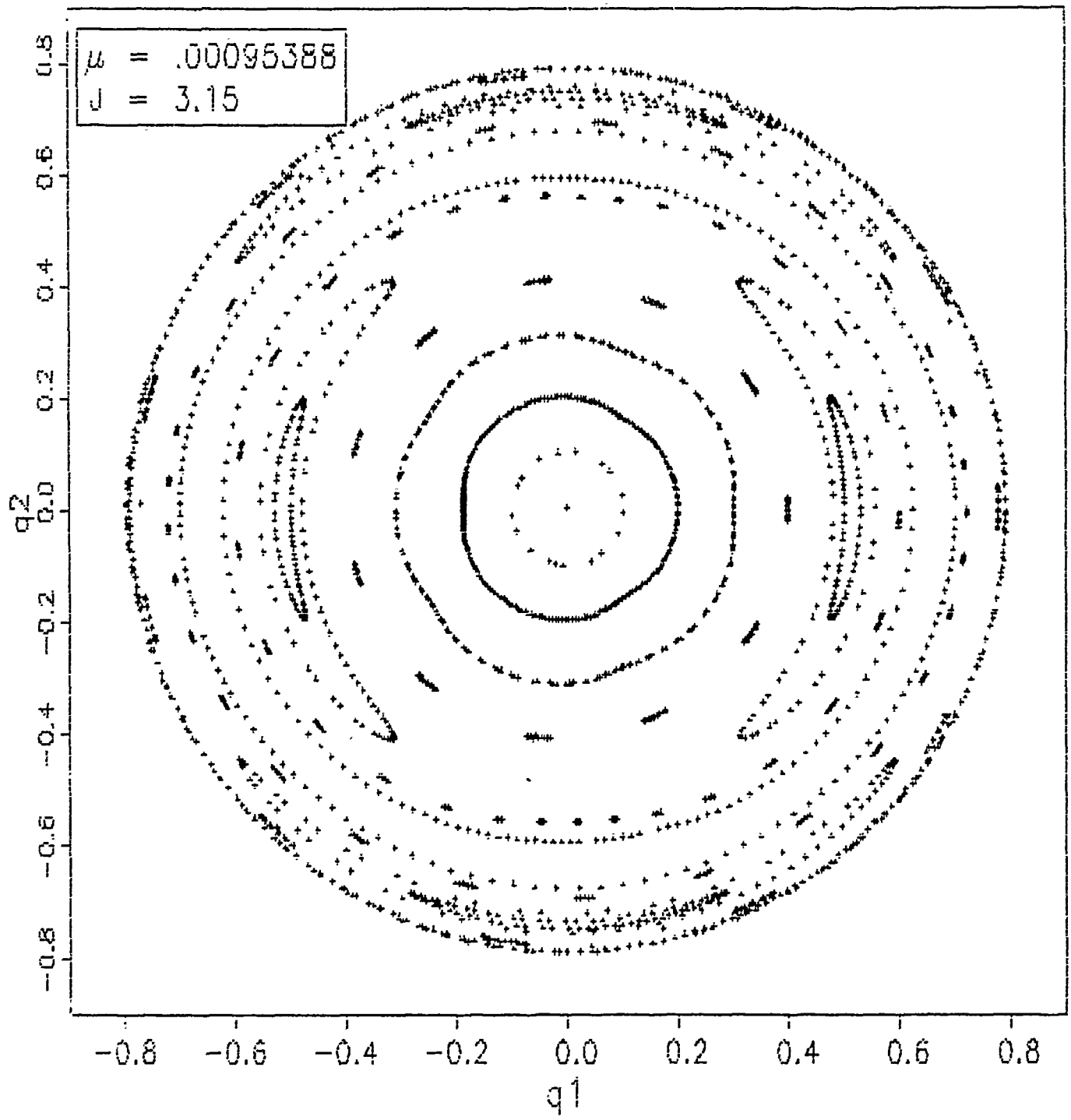


Figure 5. The Surface of Section For the Sun-Jupiter System

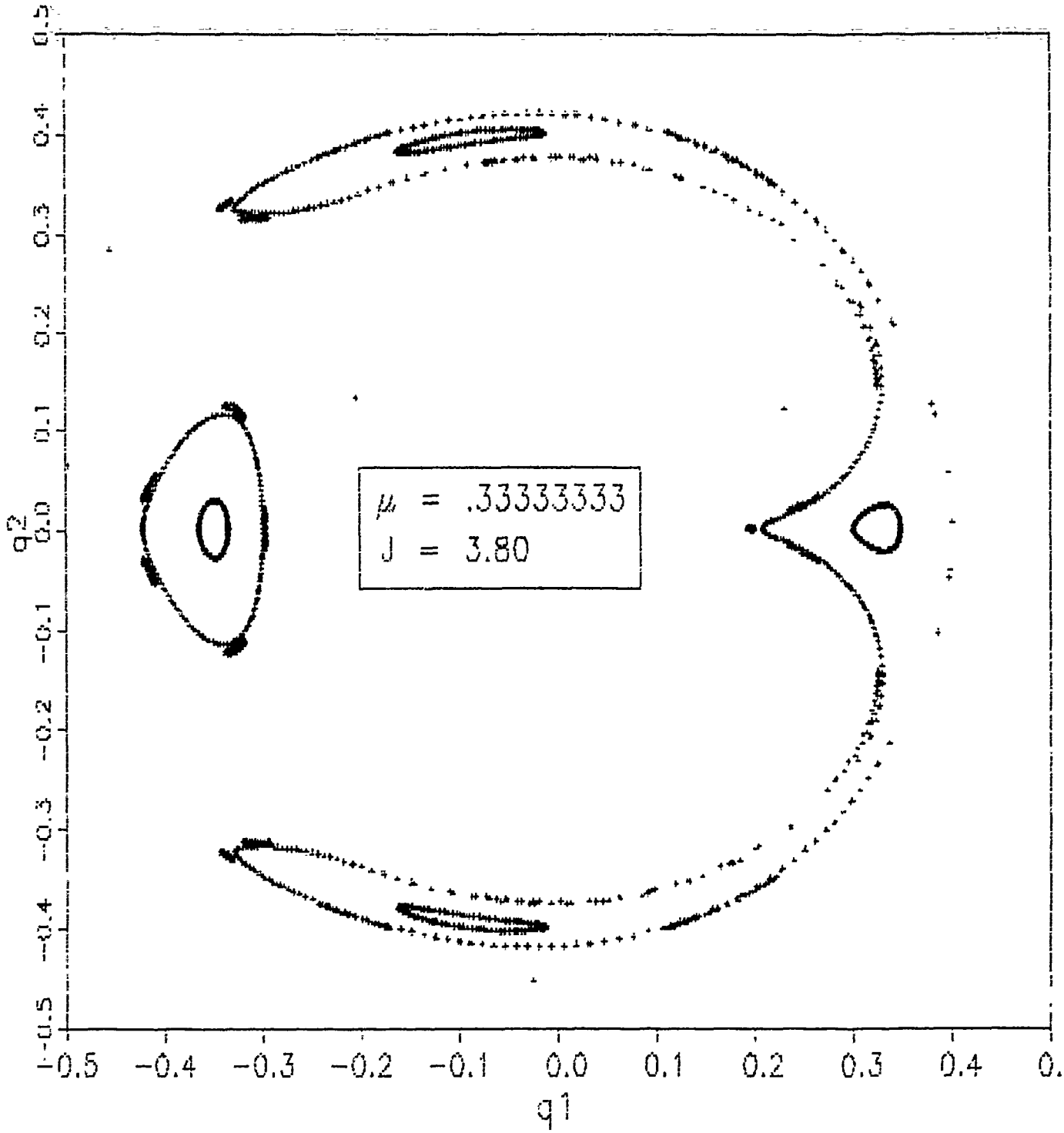


Figure 6. The Surface of Section For a Highly Perturbed System

Figures 7 and 8 illustrate these periodic regions, and their distorted appearance on a surface of section plot.

To construct an exact representation of a nearly-periodic trajectory without distortion, a vector space that is locally tangent to the Hamiltonian surface must be constructed. To this end, Wiesel introduces two variable transformations (7:233). The first, translates the frame of reference so that it is centered on the periodic trajectory. The second, termed the modal transformation, orients the frame with the local tangent space. The resulting transformation maps a second state variable into a second constant. This new constant refers to the orbit epoch time, and may be treated like the Hamiltonian constant (7:236).

These variables transformations will be used in two different ways. First, an orbit will be integrated in the original cartesian coordinates, and then transformed into the modal variables. A plot of this result will be used as an exact representation of a nearly-periodic orbit. A wide range of displacements will be tested in order to obtain a rough idea of the limits of the two-integral region.

Second, the original Hamiltonian will be canonically transformed into the modal coordinates and expanded in a Taylor's series. Since the magnitudes of the modal variables are typically much smaller than one dimensionless length, the higher order terms in the expansion rapidly approach zero. For this reason, the Hamiltonian expansion will be truncated after four terms. An approximation for the equations of motion for nearly-

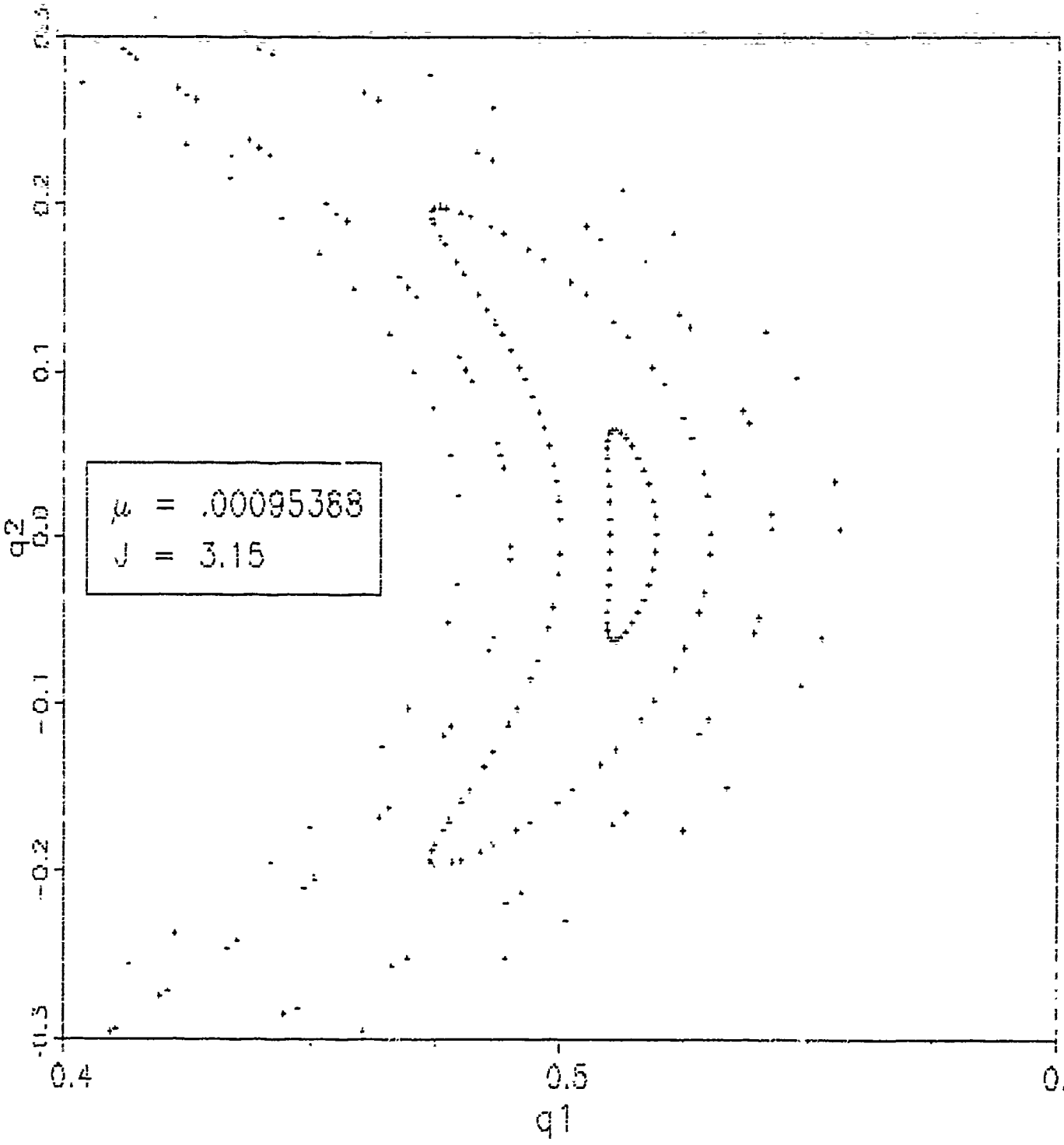


Figure 7. Magnified Periodic Region of Sun-Jupiter SOS Plot

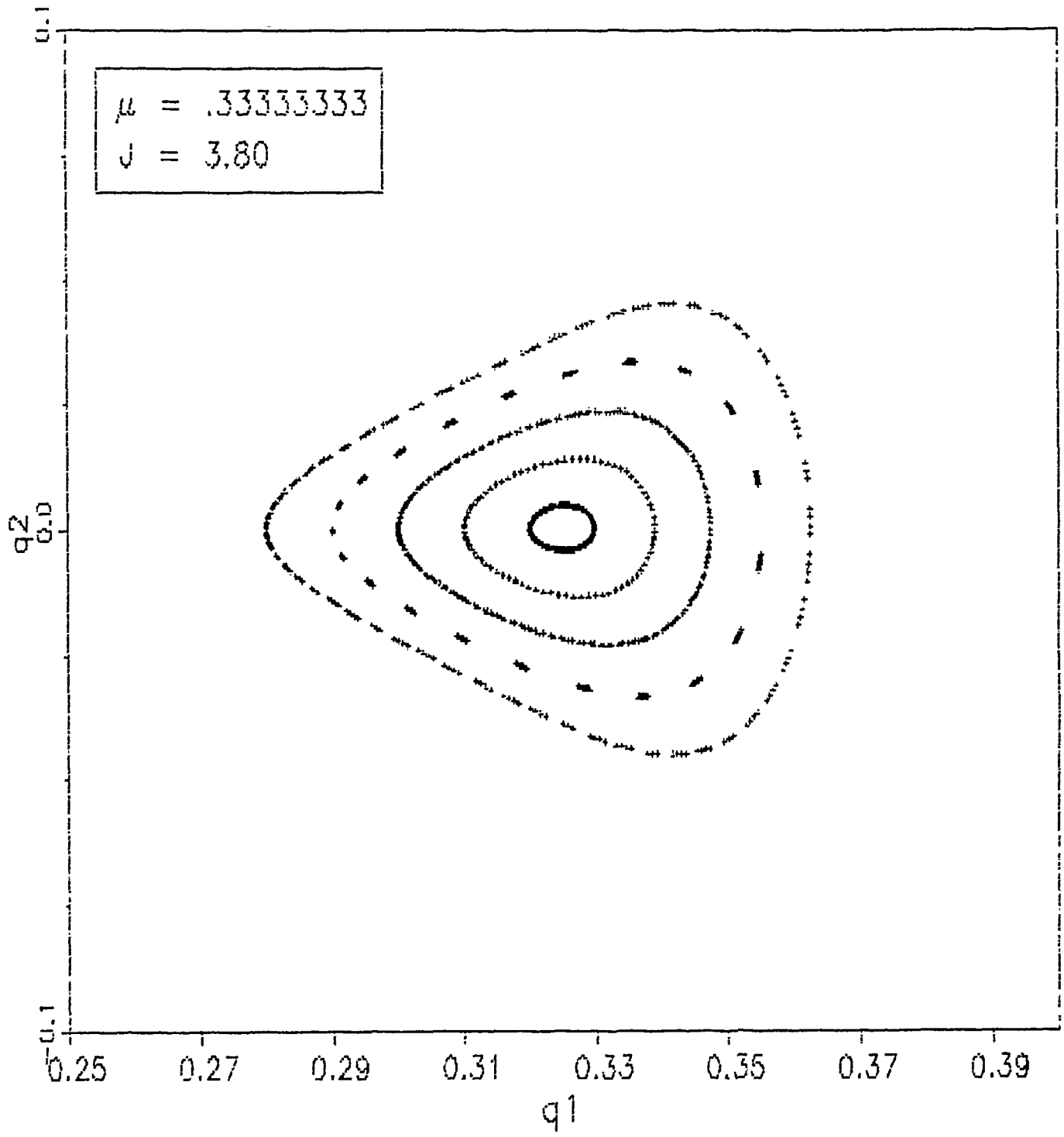


Figure 8. Magnified Periodic Region of Highly Perturbed System SOS Plot

periodic orbits will result. The validity of the expanded equations can then be tested vs. the exact representation.

The first transformation begins by allowing

$$\delta\bar{\chi} = \begin{bmatrix} \delta q_1 \\ \delta p_1 \\ \delta q_2 \\ \delta p_2 \end{bmatrix} = \begin{bmatrix} q_1 - q_{10} \\ p_1 - p_{10} \\ q_2 - q_{20} \\ p_2 - p_{20} \end{bmatrix} \tag{49}$$

to constitute a canonical transformation to a set of nearly periodic variables, centered about a periodic trajectory. Here, the zero subscript refers to the periodic reference trajectory, and the delta prefix signifies the new, nearly-periodic variables. A generating function F_2 exists

$$F_2 = (\delta p_1 + p_{10})(q_1 - q_{10}) + (\delta p_2 + p_{20})(q_2 - q_{20}) \tag{50}$$

such that it's partial derivatives of the form

$$p_i = \frac{\partial F_2}{\partial q_i} \qquad \delta q_i = \frac{\partial F_2}{\partial(\delta p_i)} \tag{51}$$

yield

$$p_i = \delta p_i + P_{i0} \qquad \delta q_i = q_i - q_{i0} \tag{52}$$

which reproduce the original substitution definition, and prove the transformation canonical.

In practice, this first transformation is more difficult than it appears. In order to subtract the current nearly-periodic trajectory from the periodic one, both trajectories must exist in an approximately continuous manner, given the tiny time steps used in the numerical integration. The method employed

here, was to preserve the periodic trajectory in a finite series of Fourier coefficients. In this way, barely a hundred samples of the actual trajectory were transformed into two sets of fifty coefficients. The advantage of the Fourier representation is that the coefficients may be reassembled into the periodic orbit at any time necessary. This is why the Fourier representation is considered continuous. Details of this digital to analog conversion process are contained in the Fourier subroutine in appendix B, and also in the book by Brouwer and Clemence (1:109).

If the variables describing the motion of the third body relative to the periodic orbit are indeed small, then Wiesel argues that the equations of motion for a nearly-periodic trajectory are analogous to the equations of variation previously described. In other words, the periodic and nearly-periodic trajectories are close enough that there is a linear relationship between the two orbits. Thus

$$\frac{d}{dt}[\delta\bar{\chi}(t)] = A(t)\,\delta\bar{\chi}(t) \tag{53}$$

is a set of linear, time periodic differential equations, and have the solution

$$\delta\bar{\chi}(t) = \Phi(t)\;\delta\bar{\chi}(0) \tag{54}$$

According to Floquet, since $A(t)$ is time periodic, then the above relation may be decomposed to

$$\delta\bar{\chi}(t) = F(t)\;e^{Mt}\,\delta\bar{\chi}(0) \tag{55}$$

where M is a constant matrix, and $F(t)$ is periodic with the same

period τ as the original orbit. Wiesel then defines the second transformation by

$$\delta\bar{\chi}(t) = F(t)\,\bar{b}(t) \tag{56}$$

where $b(t)$ is the product of e^{Mt} and an initial constant. Because e^{Mt} are the time-varying analogs of the system eigenvalues, and $F(t)$ is constructed with the system eigenvectors, this transformation is known as the modal transformation (2:671). The new state vector $b(t)$ is then obtained by simple matrix inversion.

$$\bar{b}(t) = F(t)^{-1}\,\delta\bar{\chi}(t) \tag{57}$$

The orientation of the unit vectors in the modal system is shown in figure 9. The matrix differential equation describing the periodic changes in the eigenvector matrix is

$$\frac{d}{dt}F(t) = A(t)F(t) - F(t)J(t) \tag{58}$$

where $A(t)$ is the same as in eq(31), and J is a matrix of the two system Poincaré exponents.

$$J = \begin{bmatrix} Re_1 & Im_1 & 0 & 0 \\ -Im_1 & Re_1 & 0 & 0 \\ 0 & 0 & Re_2 & Im_2 \\ 0 & 0 & -Im_2 & Re_2 \end{bmatrix} \tag{59}$$

More information on the subtleties of Floquet theory is provided by Calico (2:672). The details proving the second transformation canonical are outlined by Wiesel (7:234), where the reader is

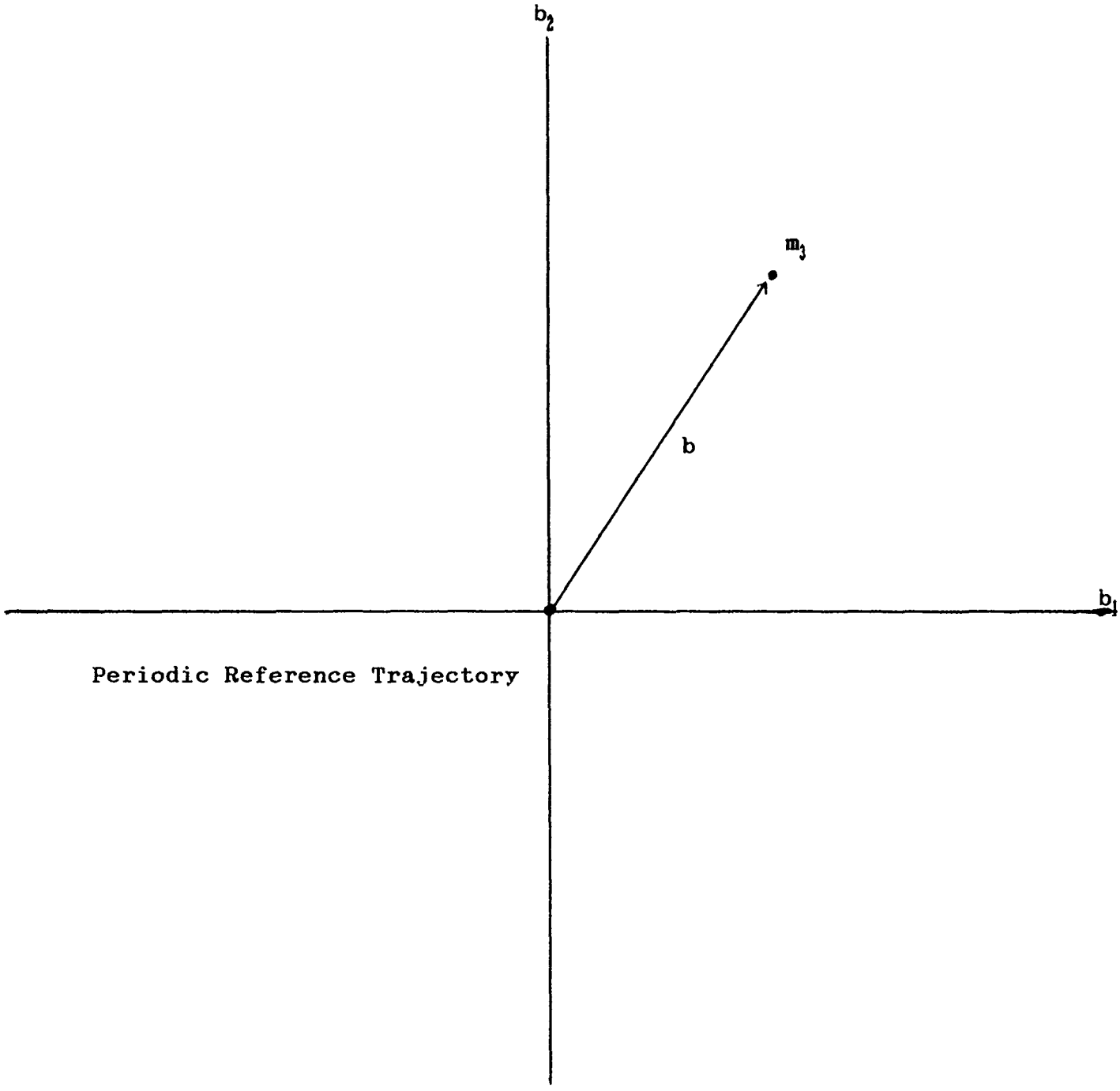


Figure 9. Reference Frame in Modal Coordinates

referred to the text by Pars (4:453-483), for a full explanation and proof.

The Restricted Three-Body Perturbation Solution

After canonical transformation to the modal coordinates, the new Hamiltonian minus a pure function of time is

$$K(\bar{b}) = H(\bar{b}) \tag{60}$$

since

$$\frac{\partial F_2}{\partial t}$$

doesn't contribute to the equations of motion.
Expanding the new Hamiltonian in a Taylor's series produces

$$K(\bar{b}) = H(0) + \sum_{i=1}^{4} \frac{\partial H(\bar{b})}{\partial b_i}\bigg|_{\bar{b}=0} b_i + \frac{1}{2!}\sum_{j=1}^{4}\sum_{i=1}^{4} \frac{\partial^2 H(\bar{b})}{\partial b_i \partial b_j}\bigg|_{\bar{b}=0} b_i b_j + \ldots . \tag{62}$$

where b=0 centers the expansion about the periodic trajectory. Alternatively, this expansion may be more compactly written using tensor notation.

$$K(\bar{b}) = H(0) + H_i(0)\,b_i + \frac{1}{2!}H_{ij}(0)\,b_i b_j + \frac{1}{3!}H_{ijk}(0)\,b_i b_j b_k + \ldots . \tag{63}$$

The first term in the expansion is the Hamiltonian for a periodic orbit, and is a constant. The second, or linear term is identically zero, because it describes the motion of the periodic trajectory with respect to itself. The third, or quadratic term is the Floquet problem, and becomes a constant coefficient, linear system in the new variables. Since the magnitude of the

modal state vector is very small compared to one, the expansion is truncated after the fourth term.

The first two elements of the modal state vector, b_1 and b_2, are the only two variables in the modal space. Elements b_3 and b_4, are both constants, and represent a change in the orbit epoch (b_3), and a change in the Hamiltonian surface (b_4). Since we are free to arbitrarily choose both of these values, they are both set to zero. The dimension of the new Hamiltonian is then reduced from four to two.

$$K(\bar{b}) = H(0) + \frac{1}{2}\,\bar{b}^T \begin{bmatrix} \omega & 0 \\ 0 & \omega \end{bmatrix} \bar{b} + \frac{1}{6}\, H_{ijk}(0)\, \bar{b}_i\, \bar{b}_j\, \bar{b}_k \qquad (64)$$

where ω is the imaginary portion of the non-zero Poincaré exponent, and the modal state vector is

$$\bar{b} = \begin{bmatrix} b_1 \\ b_2 \end{bmatrix}$$

Expanding the third-order tensor, the new Hamiltonian becomes

$$K(\bar{b}) = H(0) + \frac{1}{2}\,(b_1^2+b_2^2)\,\omega + b_1^3\alpha_1 - b_1^2 b_2\alpha_2 + b_1 b_2^2\alpha_3 - b_2^3\alpha_4 \qquad (66)$$

where

$$\alpha_1(t) = \frac{1}{6} H_{ijk}\Lambda_{1i}\Lambda_{1j}\Lambda_{1k}$$

$$\alpha_2(t) = \frac{1}{6}[H_{ijk}\Lambda_{2i}\Lambda_{1j}\Lambda_{1k} + H_{ijk}\Lambda_{1i}\Lambda_{2j}\Lambda_{1k} + H_{ijk}\Lambda_{1i}\Lambda_{1j}\Lambda_{2k}]$$

$$\alpha_3(t) = \frac{1}{6}[H_{ijk}\Lambda_{1i}\Lambda_{2j}\Lambda_{2k} + H_{ijk}\Lambda_{2i}\Lambda_{1j}\Lambda_{2k} + H_{ijk}\Lambda_{2i}\Lambda_{2j}\Lambda_{1k}]$$

$$\alpha_4(t) = \frac{1}{6} H_{ijk}\Lambda_{2i}\Lambda_{2j}\Lambda_{2k}$$

which are periodic functions of time alone. H_{ijk} and Λ_{ij} are functions of the periodic trajectory and are independent of the modal variables. Λ_{ij} represents a particular eigenvector element from the eigenvector matrix. Here, the index order is reversed such that $_i$ represents a particular eigenvector, and $_j$ indicates which element from that eigenvector is needed. Using Hamilton's equations, the equations of motion for nearly periodic orbits are

$$\begin{aligned} \frac{d}{dt} b_1 &= \omega b_2 - b_1^2\alpha_2 + 2b_1 b_2 \alpha_3 - 3b_2^2\alpha_4 \\ \frac{d}{dt} b_2 &= -\omega b_1 - b_2^2\alpha_3 + 2b_1 b_2 \alpha_2 - 3b_1^2\alpha_1 \end{aligned} \tag{68}$$

IV. Hardware - Software

This study is a collection of several programs, coded in standard Fortran 77, and executed on an ELXSI 6420 Super-Mini computer. All of the software is contained within appendix B, and is made up of six different programs and their associated subroutines. The final computer outputs are a collection of data files composed of ordinate and abscissa values. The figures presented below were created by simply plotting the unformatted data values, (x,y).

V. Numerical Technique

Searching the Phase Space

Before a periodic trajectory can be extracted, periodic regions in the phase space must first be identified using program SECTION. Isolating a periodic region and a useful set of initial conditions requires a certain amount of trial and error without prior knowledge of the phase space.

Initial Conditions

Once a periodic region has been located on a surface of section plot, a set of initial conditions, (x0,y0), a value for the Hamiltonian constant, and a value for the parameter μ can be determined. The orbital period must also be estimated by observing the number of integration steps needed for the orbit to return to it's original state, and multiplying this value by the time step (time/step).

Extracting a Periodic Trajectory

Using the initial conditions found above, a periodic trajectory is extracted from the periodic region by the program PERIOD. The initial value of the orbital period must be adjusted in order to find the period that corresponds to the Hamiltonian surface of interest. There is an inverse relationship between the orbit period and the Hamiltonian surface constant. Upon convergence, the program will calculate the eigenvectors, eigenvalues, and Poincaré exponents of the linearized system.

Storing the Periodic Trajectory

In order to preserve the periodic trajectory in nearly continuous form, the state vector and eigenvector matrix previously calculated must be fed into program FLOQUET/FOURIER. This program numerically integrates the periodic trajectory, and the state transition matrix. Routine samples of the state vector and the eigenvectors of the state transition matrix are taken and placed in temporary storage. Upon completion of the integration, the stored values are fed into a Fourier conversion subroutine. One hundred of the possible thousand integration values are converted into two sets of fifty Fourier coefficients.

Storing the Periodic Hamiltonian Coefficients

The program FLOQUET/HAMILTONIAN requires the same input as program FLOQUET/FOURIER. Here, the periodic trajectory is integrated and sampled as before. The periodic Hamiltonian coefficients are then calculated by a complex series of vector multiplications. As before, the values are temporarily stored until the integration is complete. Another call to the Fourier subroutine produces the desired set of coefficients.

The Exact Nearly-Periodic Trajectory

The exact nearly-periodic trajectories are created by the program EXACT. The orbit is integrated in the original variables, before the state vector is routinely extracted and transformed into the modal variables. Three separate data files result. One each for the two constants created by the modal

transformation vs. time, and one for the remaining two variables plotted together. The only inputs required include the Fourier coefficients of the periodic orbit, and a value for the initial displacement off the periodic center. The output is an unformatted data file, whose (x,y) entries may be plotted directly.

The Expanded Nearly-Periodic Trajectory

Here, the new set of equations derived from the expanded Hamiltonian are integrated. The inputs required include the Fourier coefficients that represent the periodic Hamiltonian coefficients, and the same initial displacement used in program EXACT converted to modal coordinates. Program EXACT provides these values. The output is a data file containing both elements of the state vector, sampled during the integration.

Numerical Analysis

There are two distinct observations to made in this analysis. First, several exact trajectories will integrated and grouped according to their initial displacements from the periodic trajectory. A rough idea of the behavior of nearly-periodic orbits vs initial displacement can be obtained. Also, the validity of the two integral assumption will be monitored by plotting both constants of motion vs time.

The second set of observations will be made by overlaying several expanded orbits on top of the exact ones. This step is critical in determining the proper truncation limit for the

expanded Hamiltonian. Both sets of observations will be accomplished twice. Once for the Sun-Jupiter system, and once for the highly perturbed system.

VI. Results and Discussion

Nearly-Periodic Orbits in Modal Variables

Figure 5 is the surface of section plot for the Sun-Jupiter system. Here, μ is very small, and as one might expect, this system could well be described using classical two-body perturbation theory, were it not in the vicinity of a resonance.

> Proximity to a resonance, means that the same relative configurations repeat in the same order. This gives an otherwise small gravitational force a chance to produce a relatively large effect. (9:98)

By comparison, figure 6 represents a highly perturbed dynamical system. These are only two of an infinite number of choices, but were chosen because of the plainly visible closed curves present on both. Figures 7 and 8, magnify these regions where two integrals of motion are present.

To this point, all of the figures presented have been in the coordinate frame presented in figure 2. From this point on, all of the figures will be referenced to the modal unit vectors described in figure 9, and/or a time axis measured in orbital radians.

The exploration of nearly-periodic three-body trajectories, begins with figure 10. Here, orbits very close to the Sun in the Jupiter-Sun system are plotted. As was expected, a set of concentric circles is present. The initial displacement off the periodic trajectory for each orbit is recorded as a fraction of μ. This plot depicts the limiting region where two integrals are said to exist. Any distortion of the trajectories indicates the

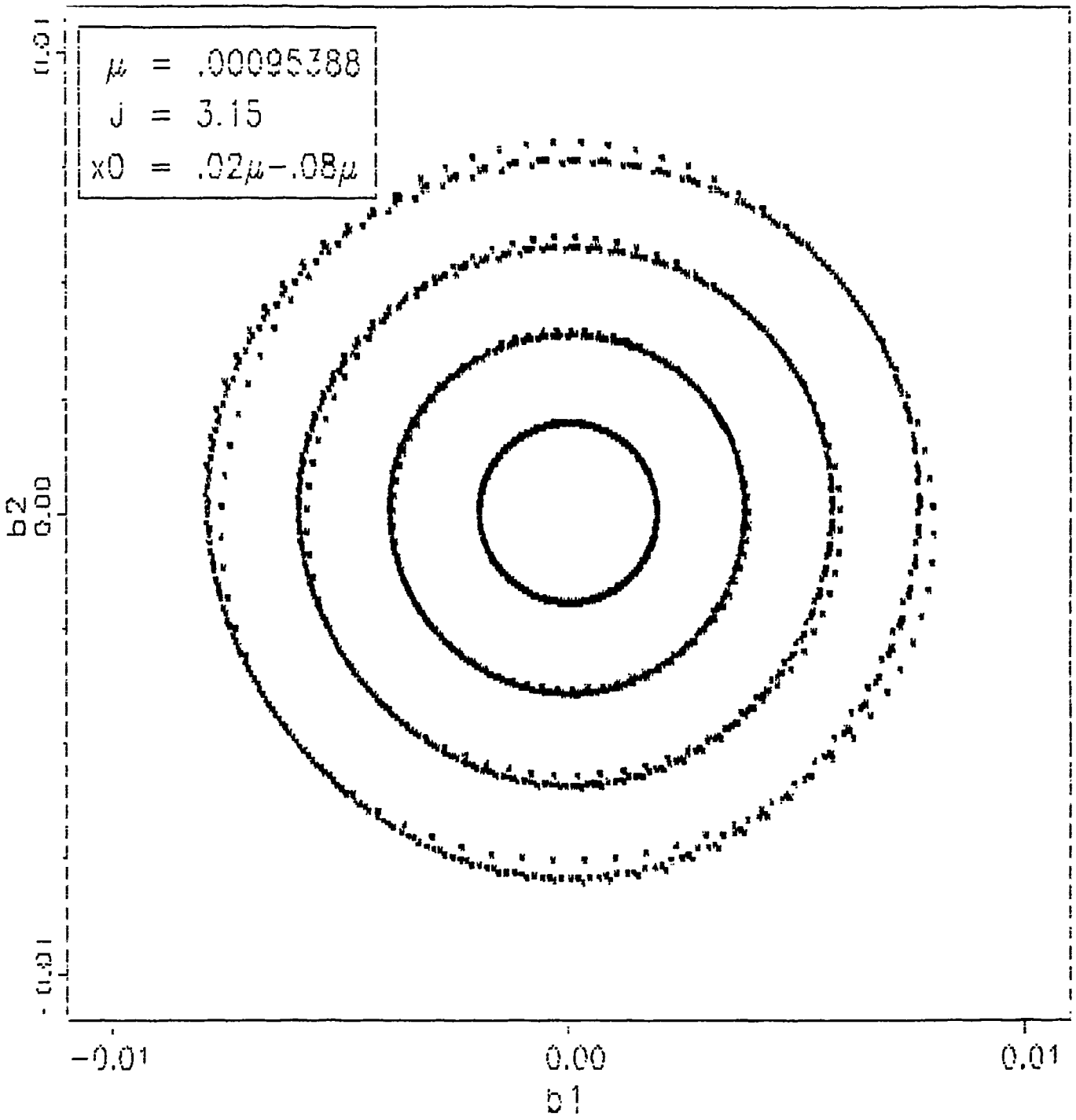


Figure 10. Nearly-Periodic Orbits in the Vicinity of Two Exact Integrals of Motion (Sun-Jupiter System)

gradual dissolution of the second constant. Figure 11 shows the behavior of the two integrals for the outer trajectory of figure 10, plotted on the same scale. The small periodic displacements on figure 11 correspond to the distortions on figure 10. Since straight lines and perfectly round circles are all that exist for the trajectories inward from this point, they were considered uninteresting and left alone.

The marginal case, where the second integral of motion can no longer be assumed constant, is pictured in figure 12. Here, the initial displacements off the periodic trajectory are increased by an order of magnitude. The circular orbits appear to decompose into five separate trajectories. Surprisingly, however, each orbit continues to close upon itself. Figure 13 shows the marginally constant nature of the second integral.

The extreme case, where Jacobi's integral is the only constant in the system, is plotted in figure 14. Unbelievably, these orbits about the sun continue to close. This is very interesting since the apogee of the orbit extends all the way out to, and even past Jupiter! Figure 15 plots the third state variable, formerly the second integral, and Jacobi's constant.

The next series of plots, figures 16-21, represent similar cases to the ones presented, only now for the system where μ is equal to a third. Once again, the first plot in the series, figure 16, illustrates the nicely concentric nature of the

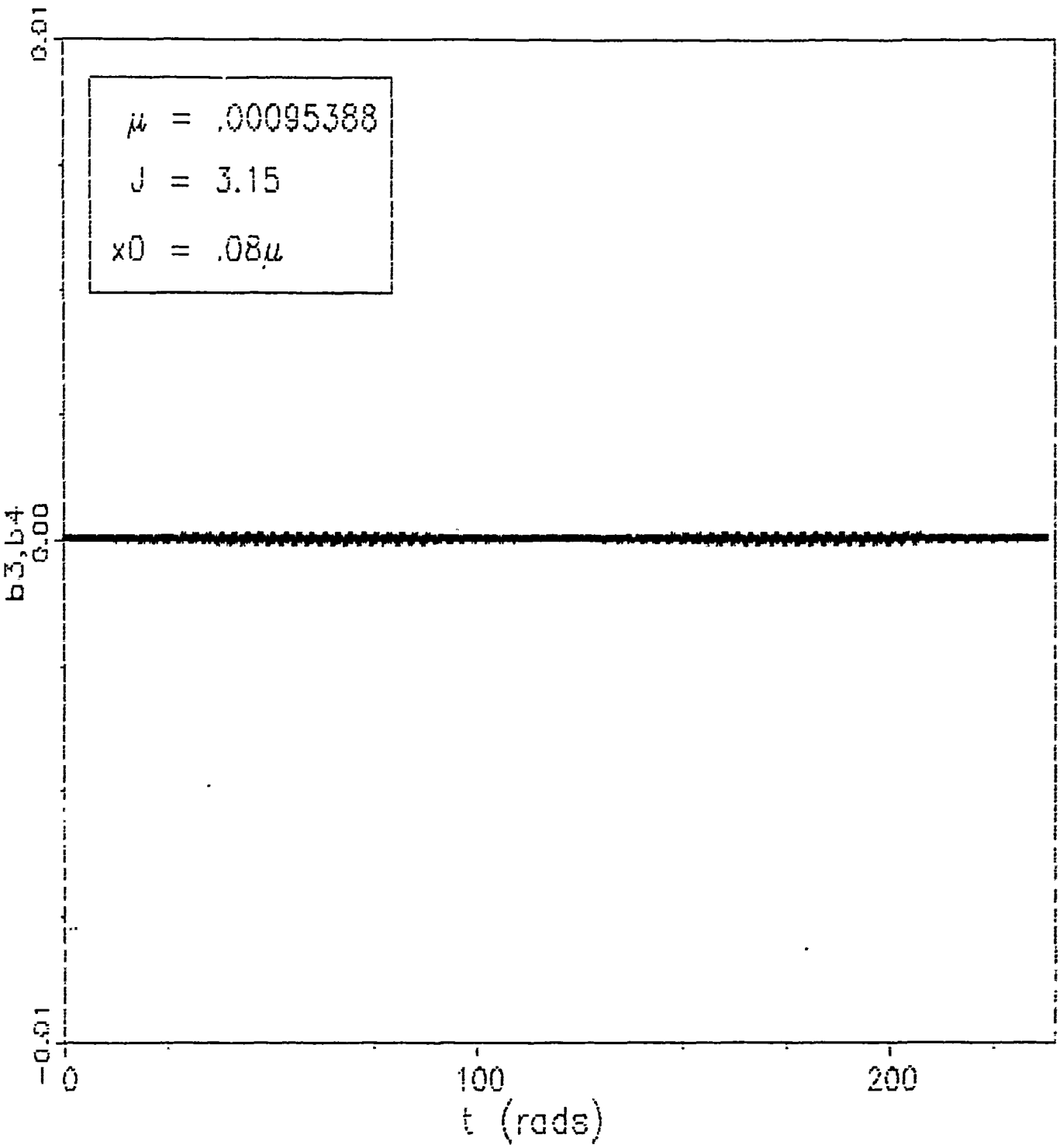


Figure 11. Epoch and Hamiltonian Constants vs. Time in the Vicinity of Two Exact Integrals of Motion (Sun-Jupiter System)

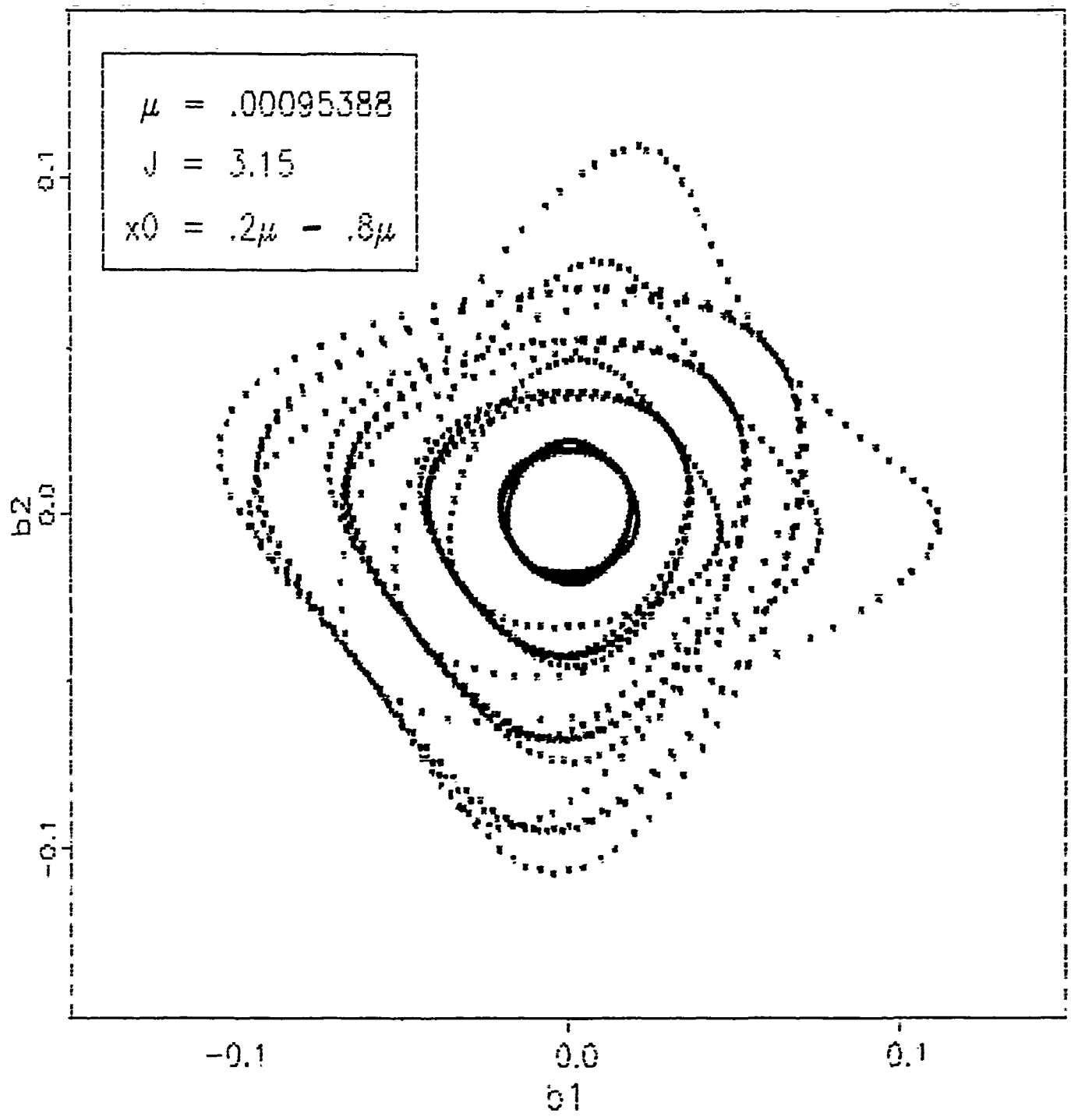


Figure 12. Nearly-Periodic Orbits in a Transition From Two Exact Integrals of Motion to One (Sun-Jupiter System)

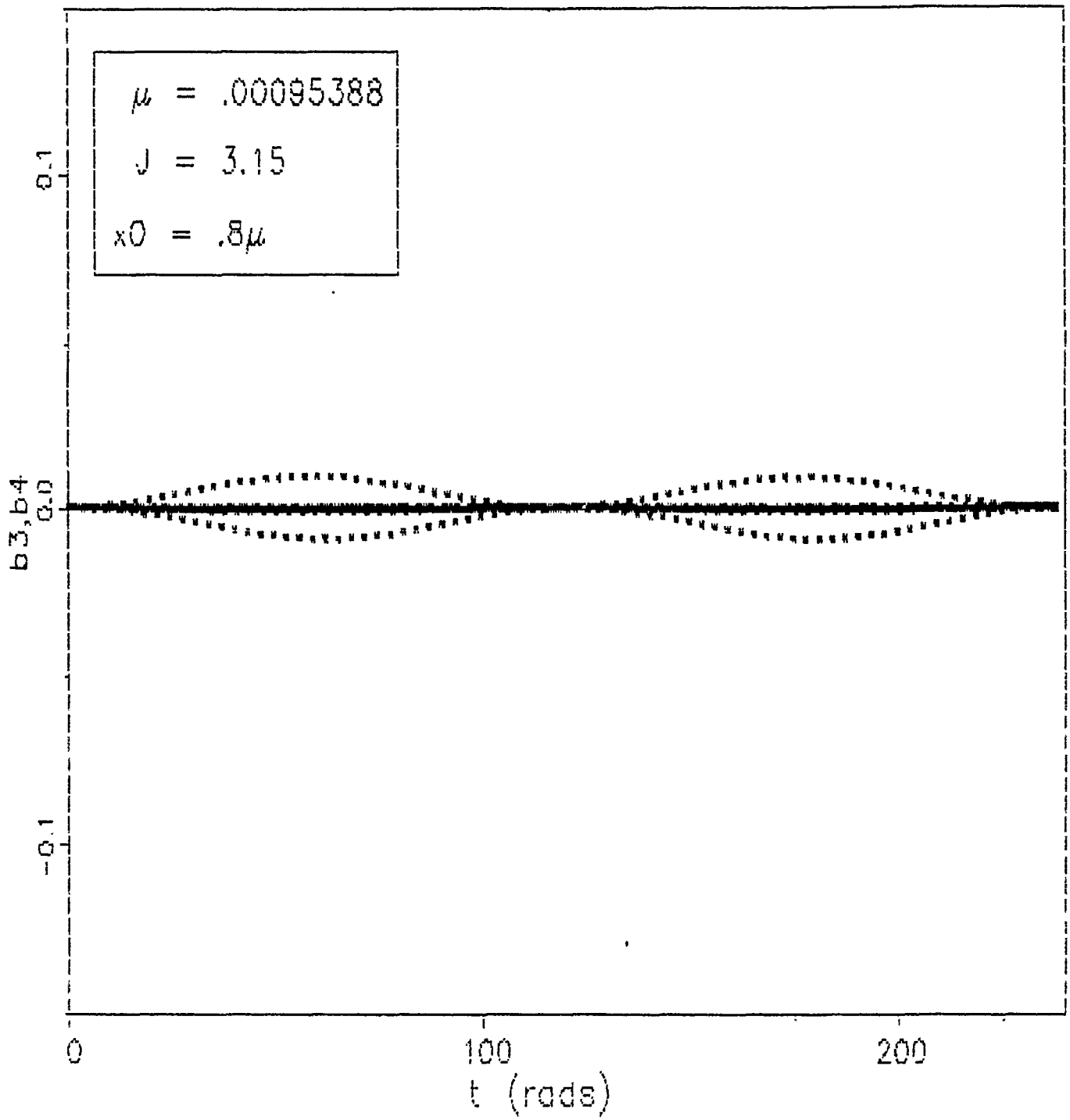


Figure 13. Epoch and Hamiltonian Constants vs. Time in a Transition From Two Exact Integrals to One (Sun-Jupiter System)

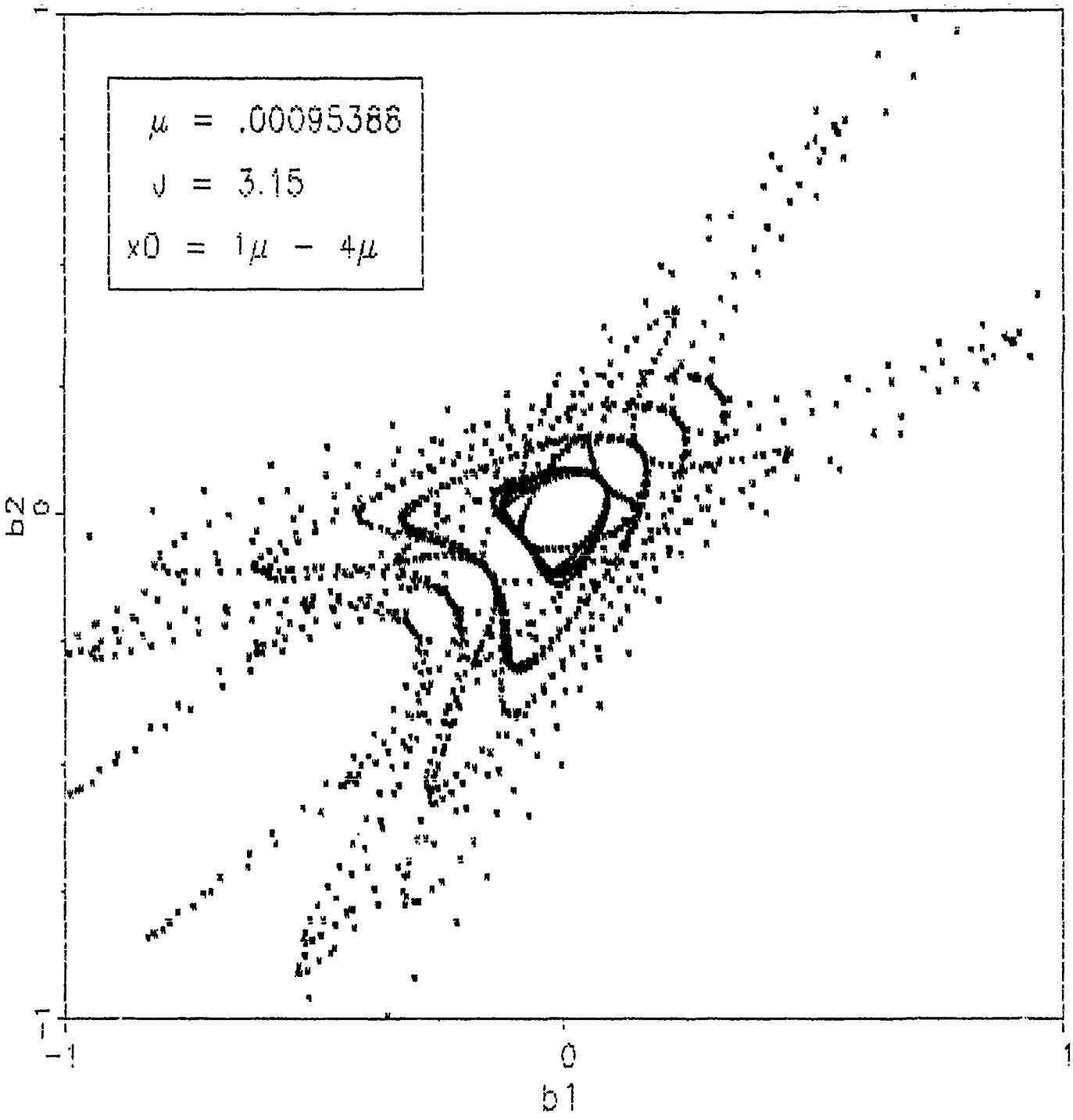


Figure 14. Nearly-Periodic Orbits in the Absence of a Second Integral of Motion (Sun-Jupiter System)

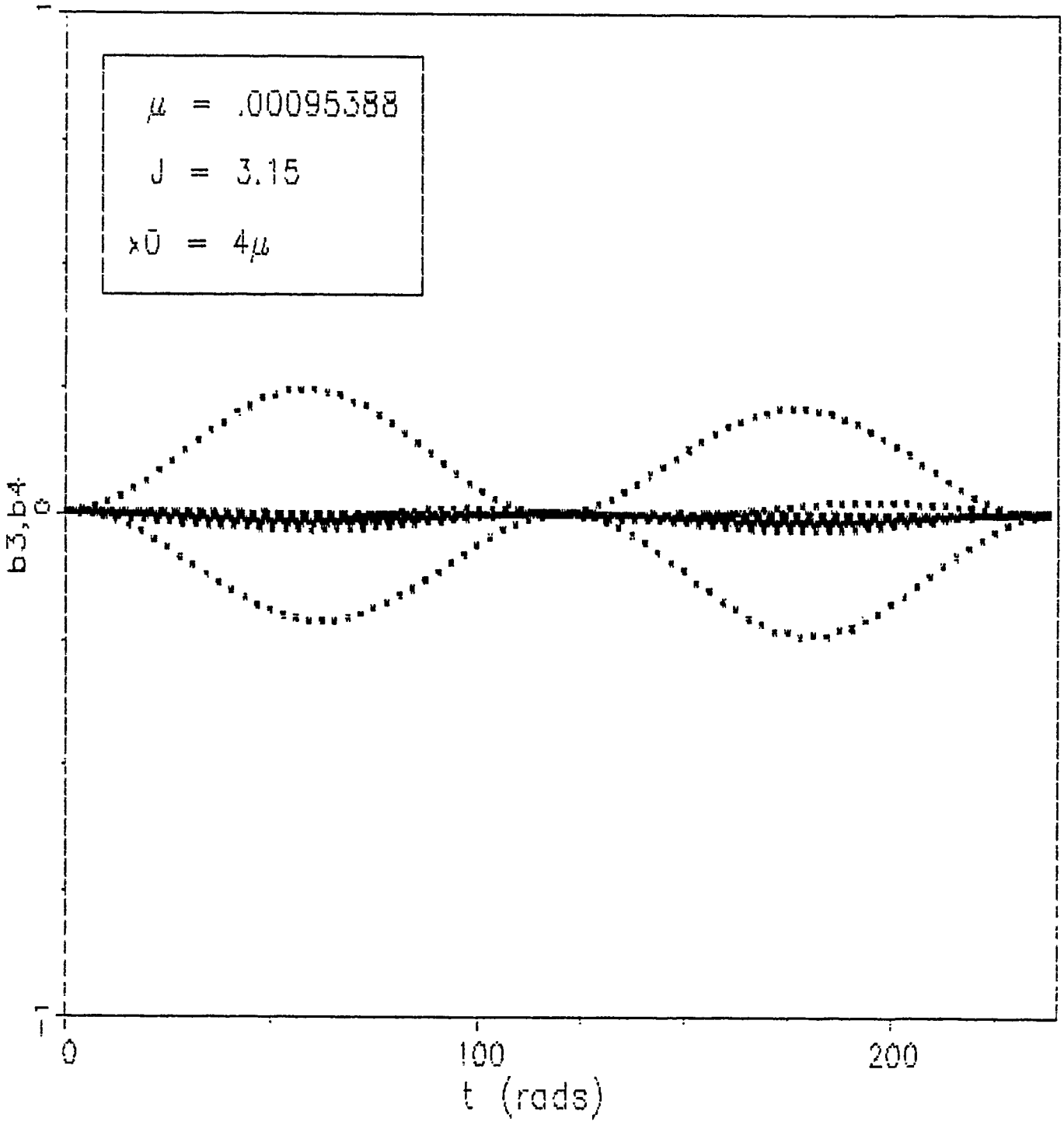


Figure 15. Epoch and Hamiltonian Constants vs. Time in the Absence of a Second Integral of Motion (sun-Jupiter System)

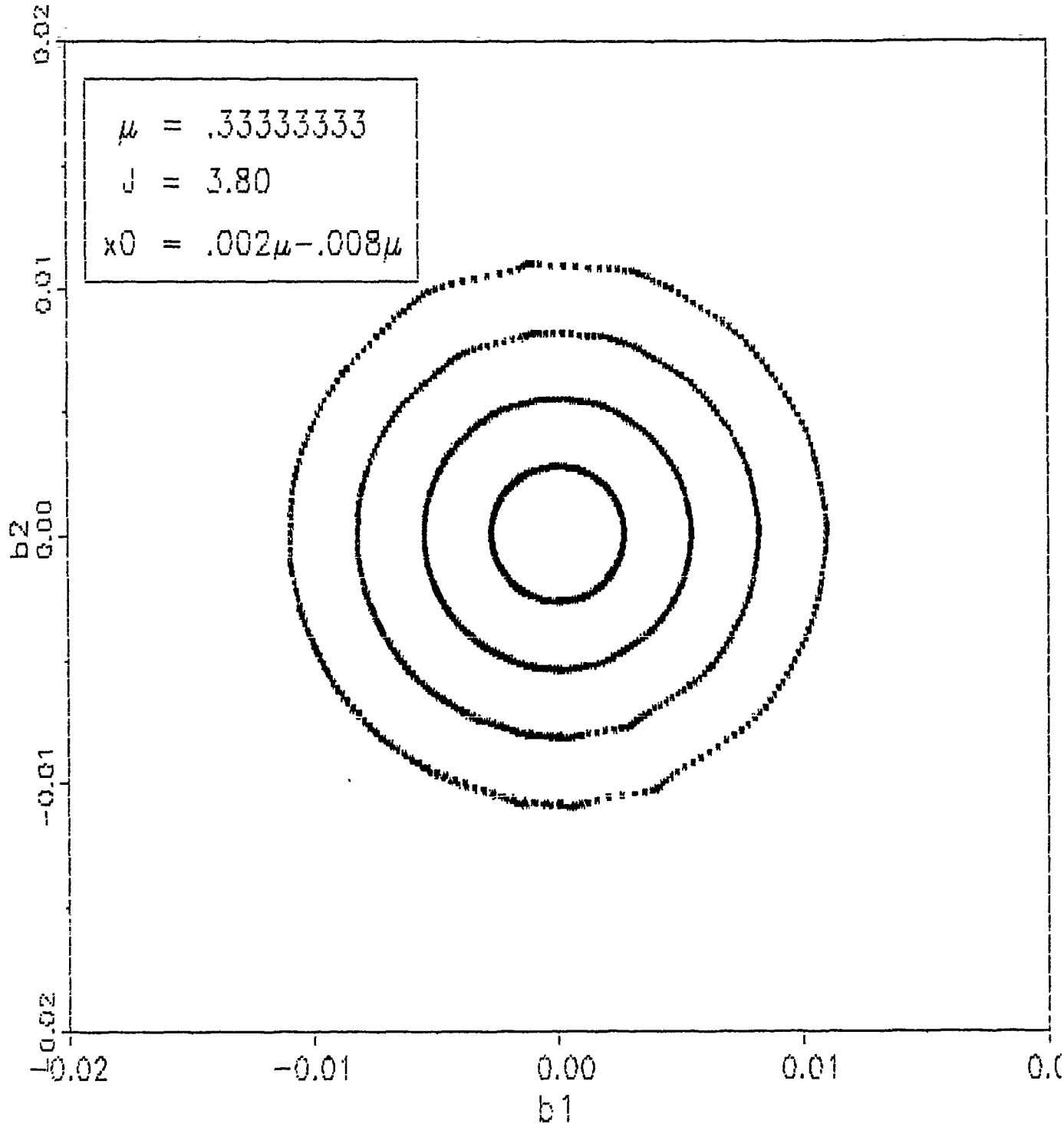


Figure 16. Nearly-Periodic Orbits in the Vicinity of Two Exact Integrals of Motion (Highly Perturbed System)

nearly-periodic orbit in the presence of both integrals. Figure 17 confirms that there are indeed two constants of motion present.

In figure 18, the phenomenon of chaotic motion is first seen. As we step further away from periodicity, the orbits develop increasing numbers of odd twists and turns. The outer three orbits fail to close at all. Figure 19 is very interesting, since for the first time, both integrals of motion appear to be moving. Since Jacobi's constant has been proven to exist everywhere in the phase space, then figure 19 indicates that the tangent space is no longer aligned with the Hamiltonian surface.

Figure 20 is an excellent example of a formerly well behaved dynamical system marching off to chaos. The inner two trajectories are still recognizable as orbits that nearly close. The outer two trajectories no longer describe an orbit. The onset of chaotic motion is well presented in figure 21. For approximately half an orbit, the second integral remains quasi constant, and the coordinate frame anchored on the Hamiltonian surface. As time progresses, the epoch constant becomes the epoch variable, and the relationship between coordinate reference and the Hamiltonian surface is destroyed. Total chaos ensues.

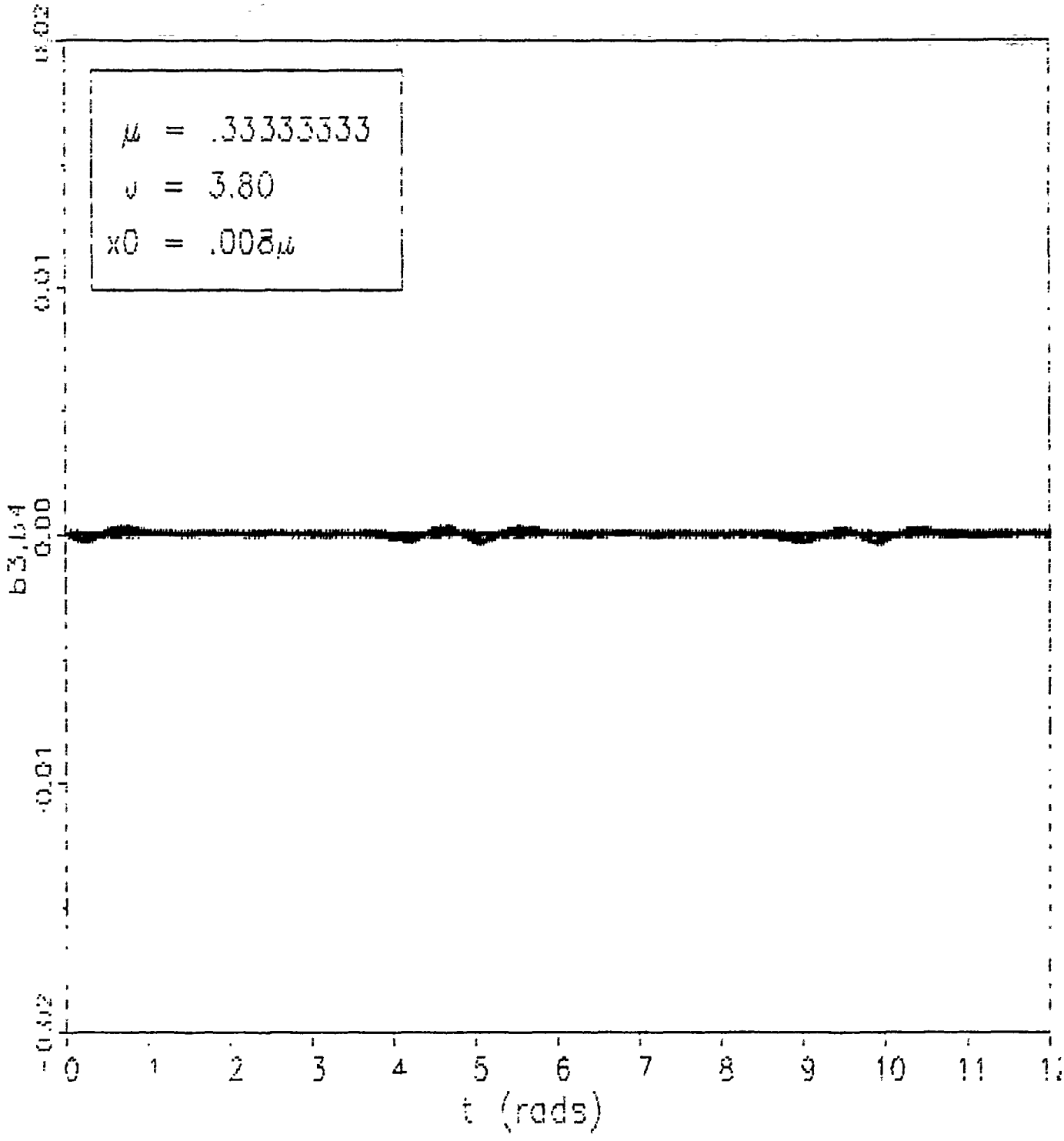


Figure 17. Epoch and Hamiltonian Constants vs. Time in the Vicinity of Two Exact Integrals of Motion (Highly Perturbed System)

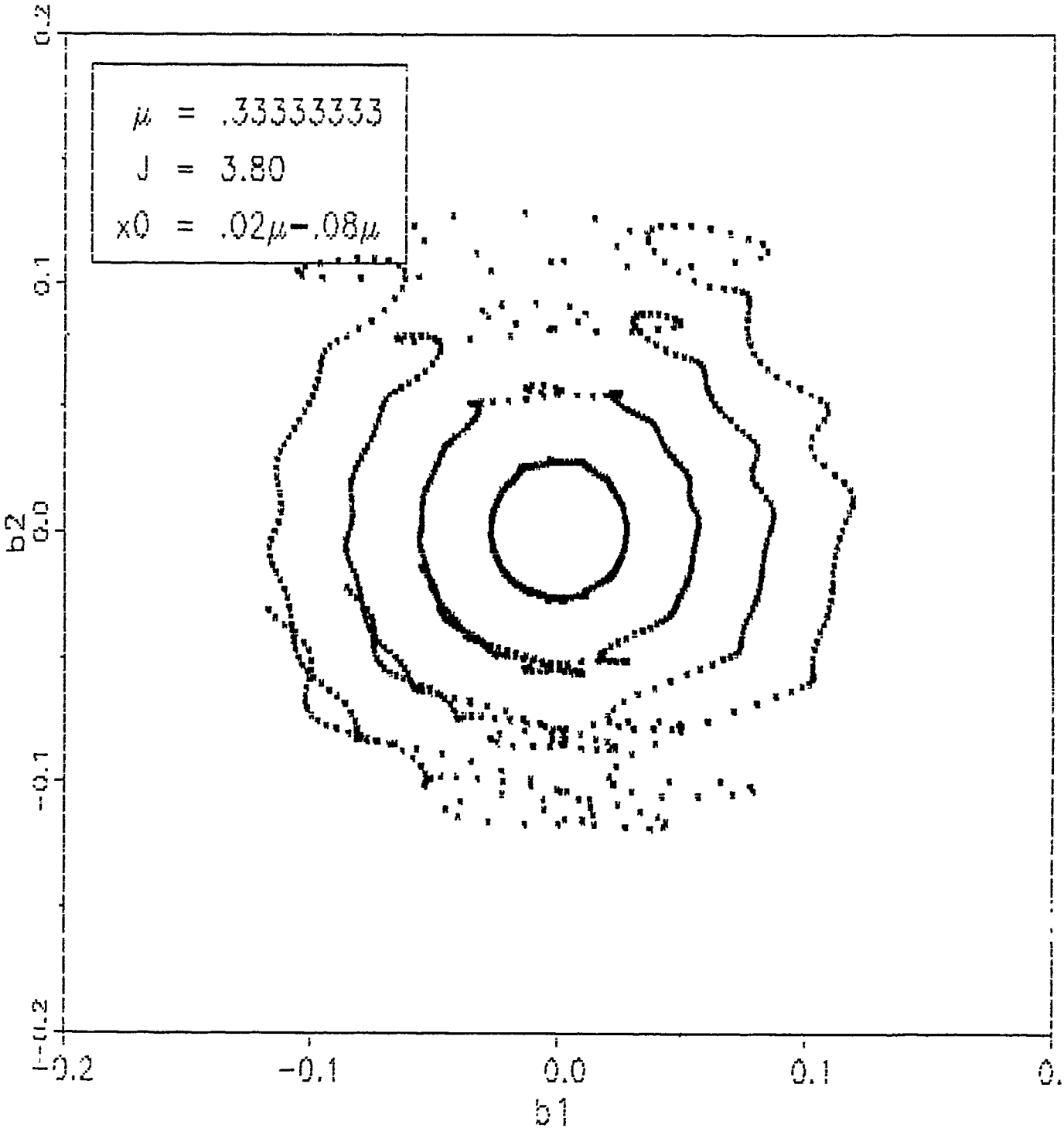


Figure 18. Nearly-Periodic Orbits in a Transition From Two Exact Integrals of Motion to One (Highly Perturbed System)

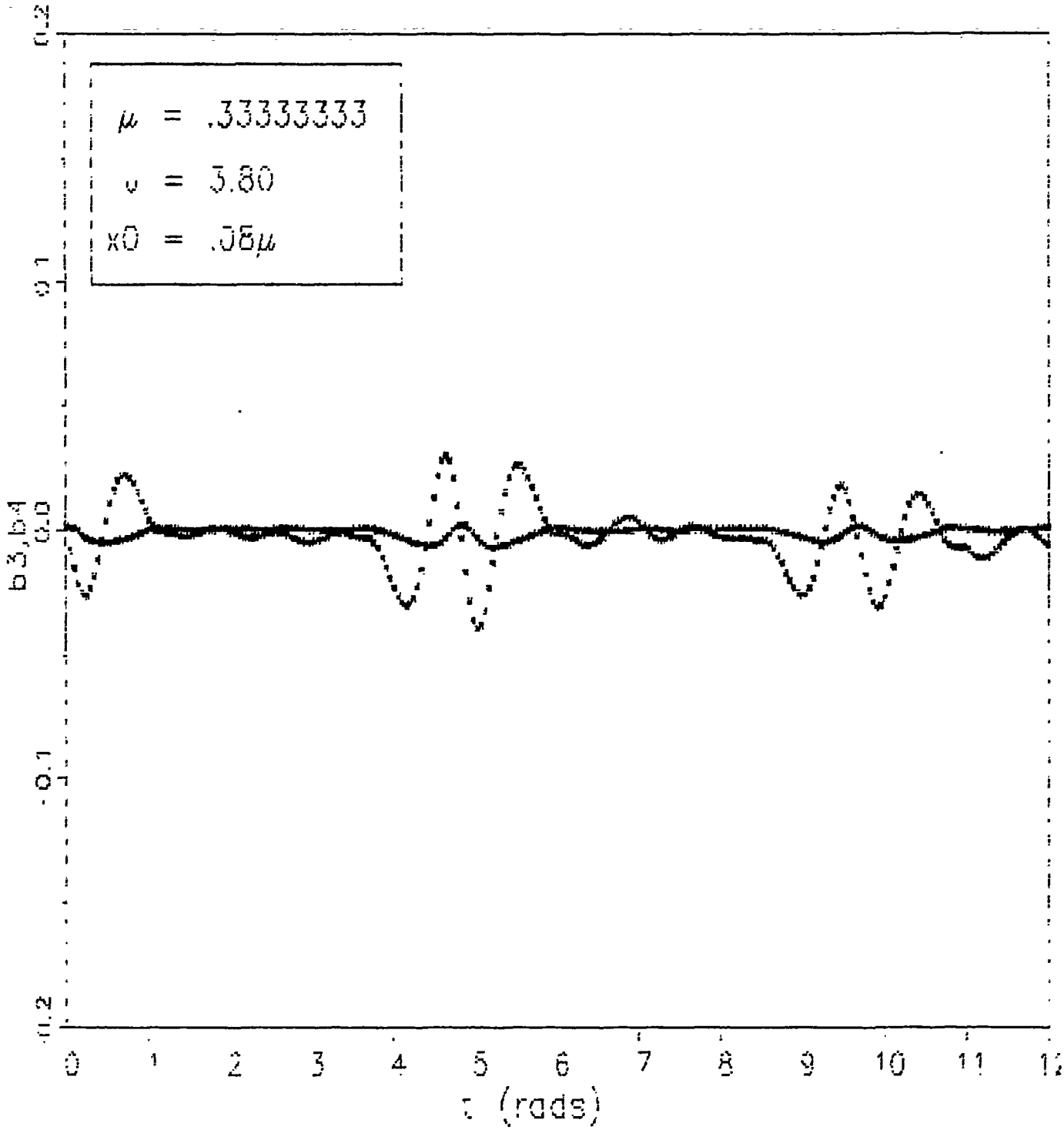


Figure 19. Epoch and Hamiltonian Constants vs. Time in a Transition From Two Exact Integrals to One (Highly Perturbed System)

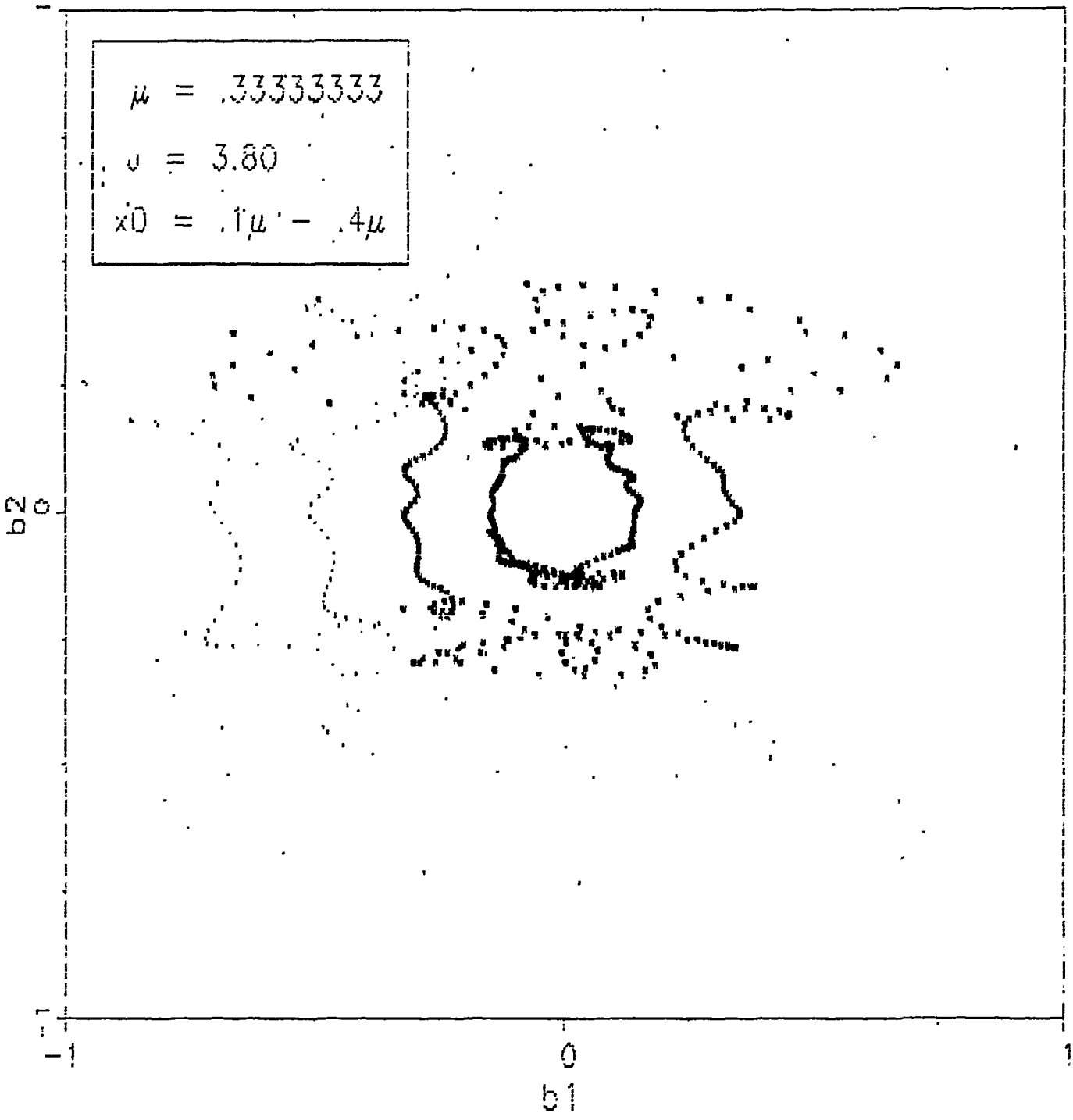


Figure 20. Nearly-Periodic Orbits in the Absence of a Second Integral of Motion (Highly Perturbed System)

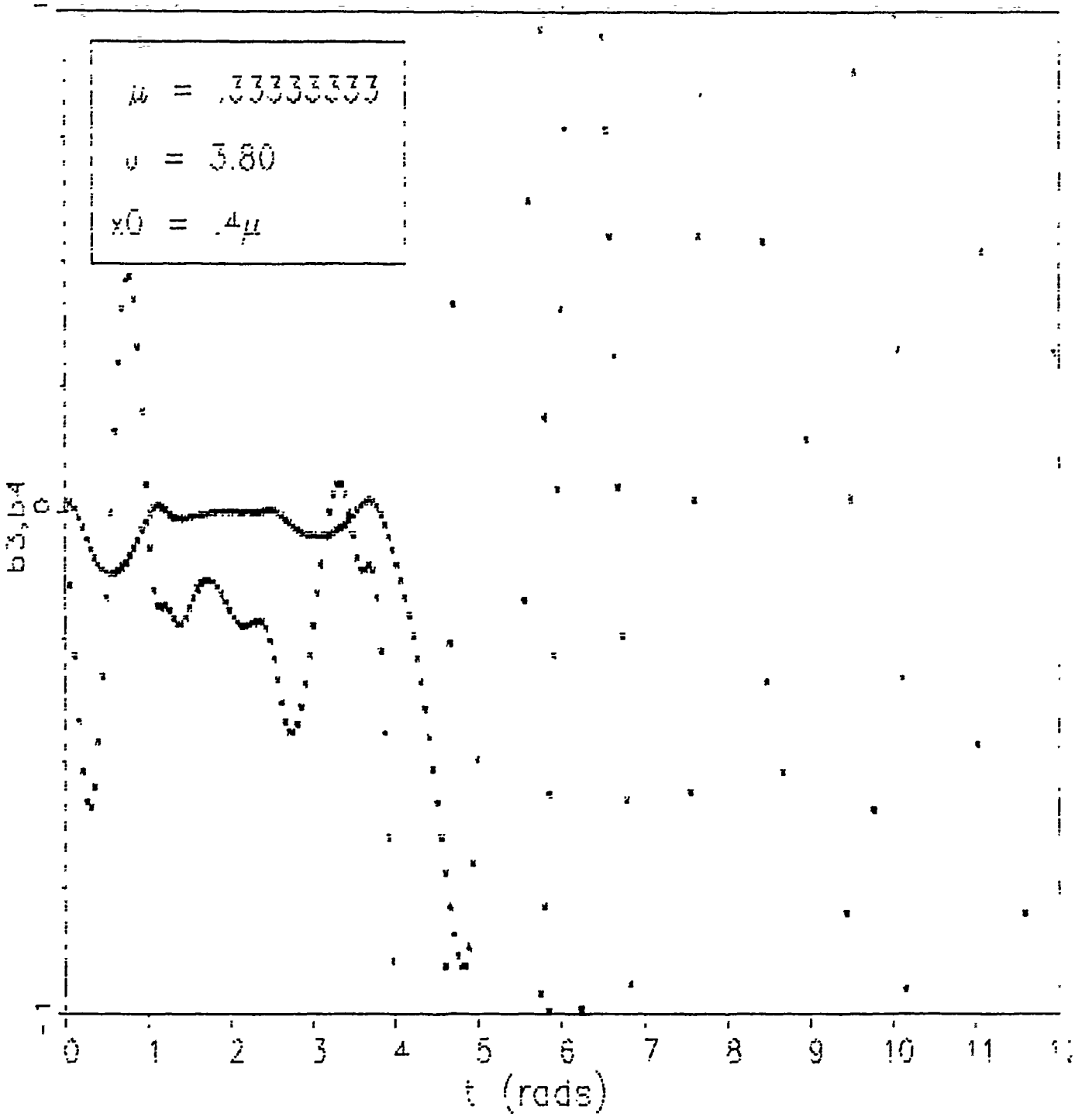


Figure 21. Epoch and Hamiltonian Constants vs. Time in the Absence of a Second Integral of Motion (Highly Perturbed System)

The Expanded Approximation Vs. The Exact Case

To best illustrate the accuracy of the approximated equations of motion, they have been overlaid with a plot of the exact case. Figure 22 compares two of the trajectories from figure 10, with the same two trajectories from the expanded case. The inner orbits overlay in round concentric fashion as expected. The outer overlay represents the limit of the highly correlated region. In figure 23, the trajectories are very marginally agreeable, while in figure 24, the trajectories only meet at nodal points.

For μ equal to a third, similar behaviors may be observed. Figure 25 represents the highly correlated region. Figure 26, however, shows a large difference between the expanded and exact cases. This case is still marginal, since both represent nearly closed orbits of the same size. In figures 27, the expanded trajectories don't show any indication of the chaotic motion present in the exact case.

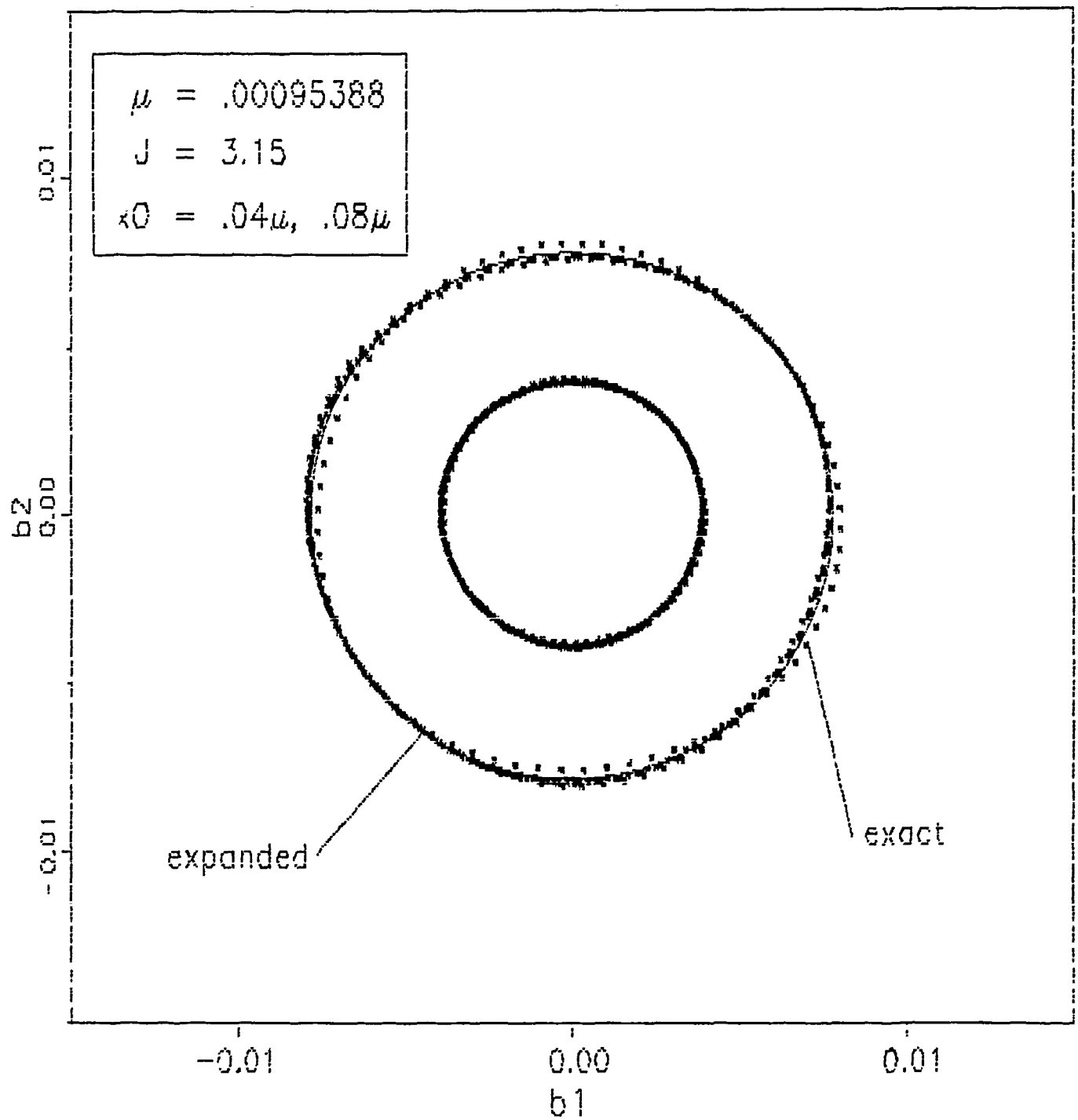


Figure 22. Overlay of Exact and Expanded Nearly-periodic Trajectories in the Vicinity of Two Integrals of Motion (Sun-Jupiter System)

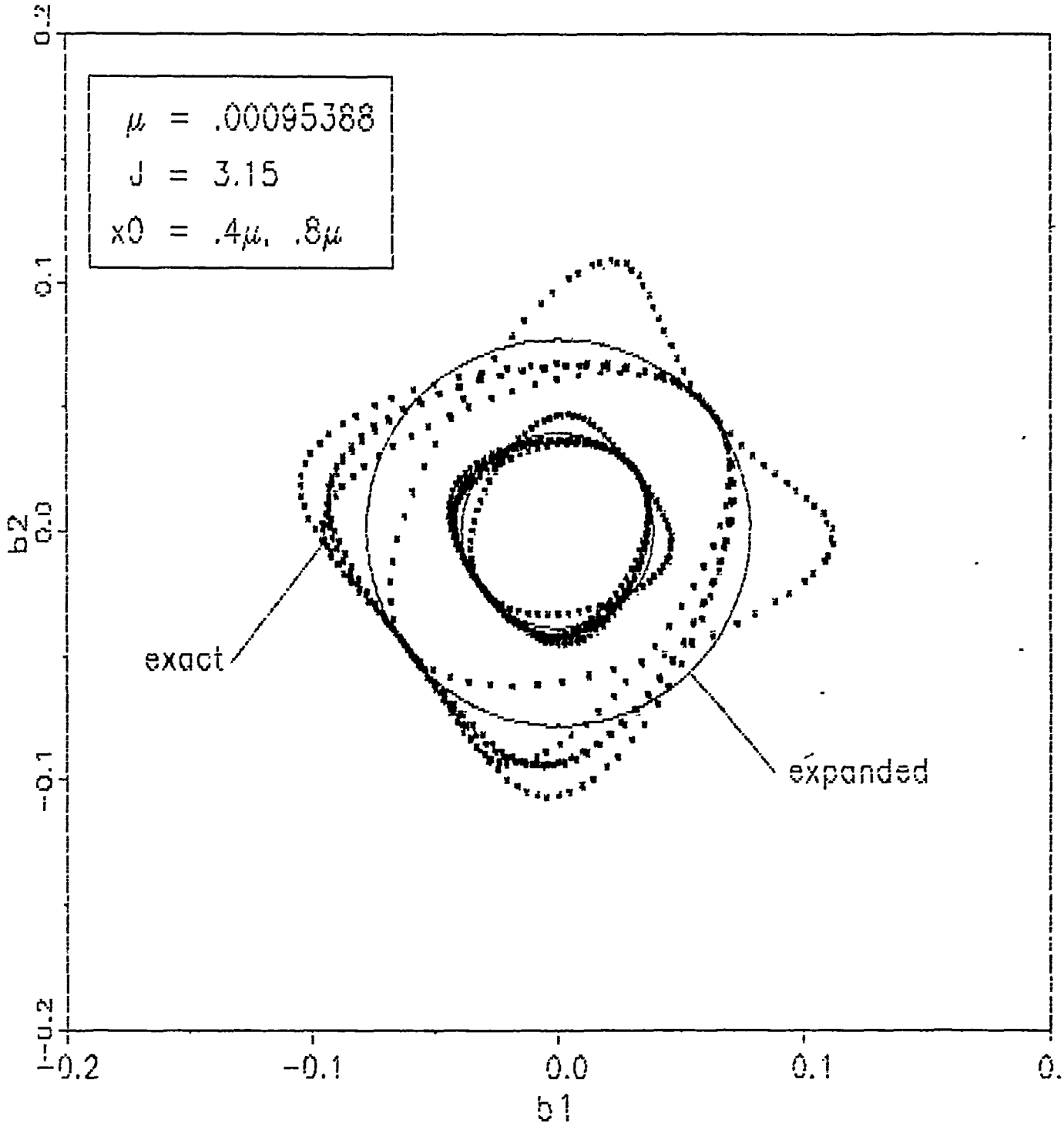


Figure 23. Overlay of Exact and Expanded Nearly-periodic Trajectories in a Transition From Two Exact Integrals of Motion to One (Sun-Jupiter System)

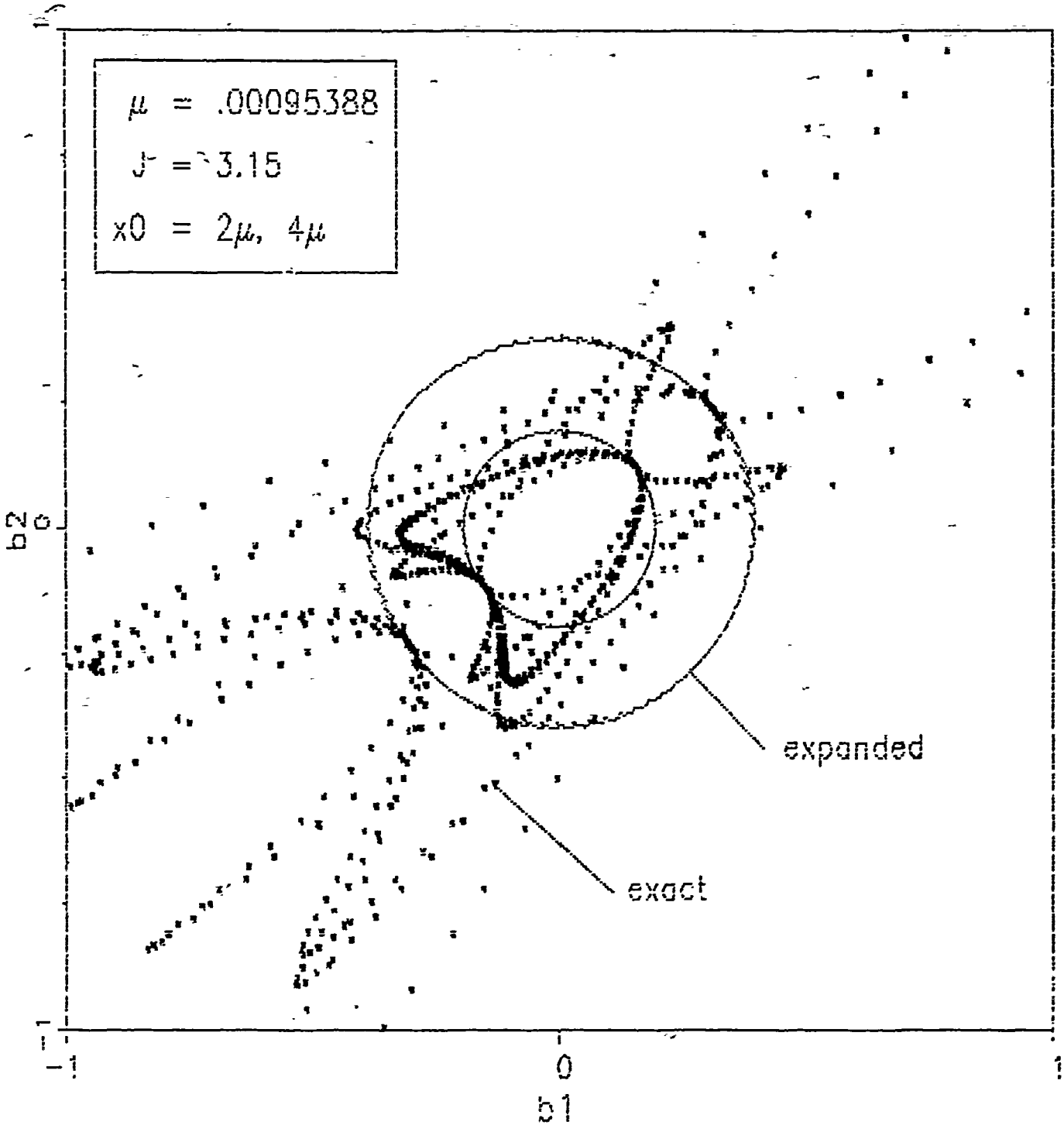


Figure 24. Overlay of Exact and Expanded Nearly-periodic Trajectories in the Absence of a Second Integral of Motion (Sun-Jupiter System)

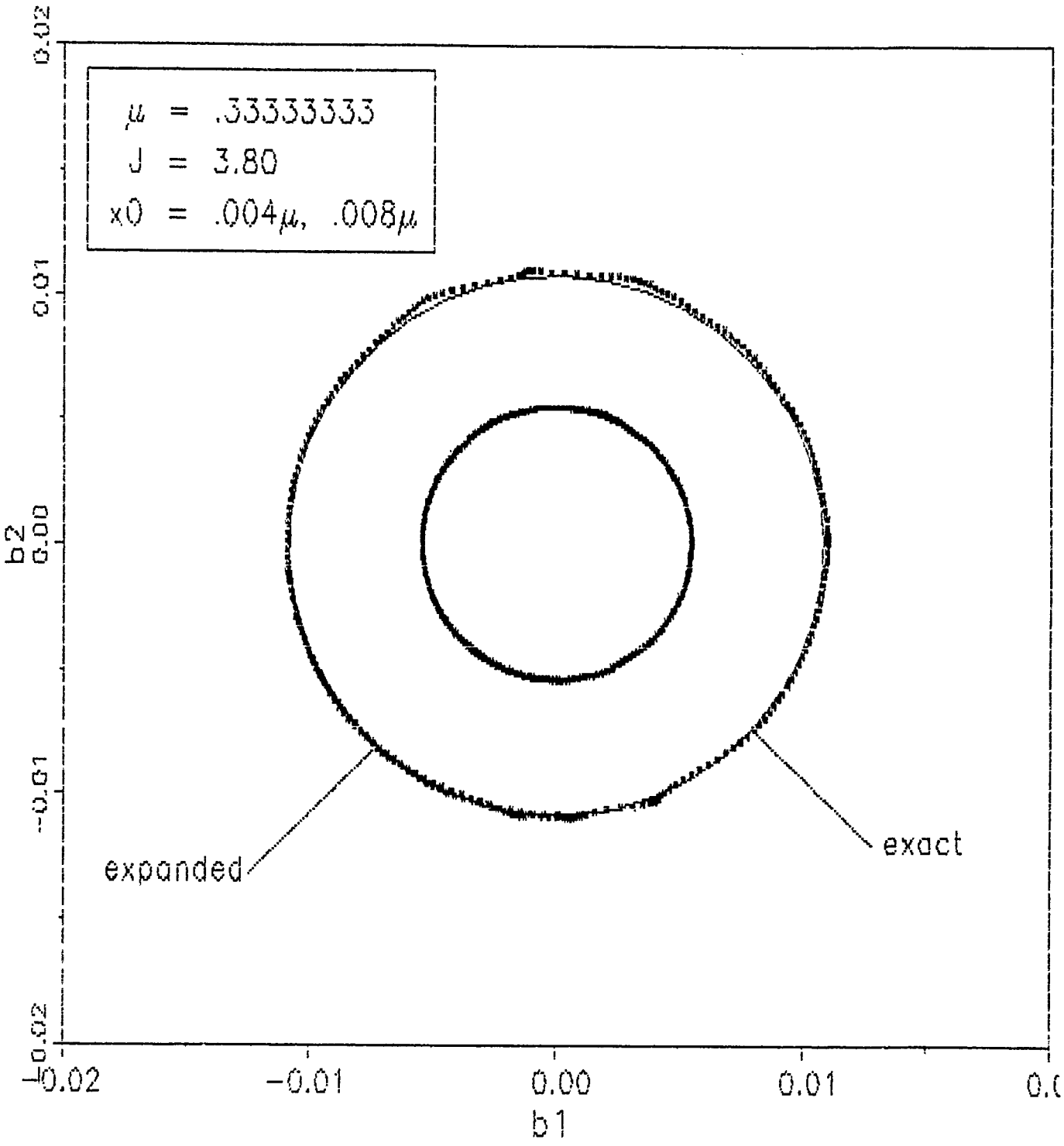


Figure 25. Overlay of Exact and Expanded Nearly-periodic Trajectories in the Vicinity of Two Integrals of Motion (Highly Perturbed System)

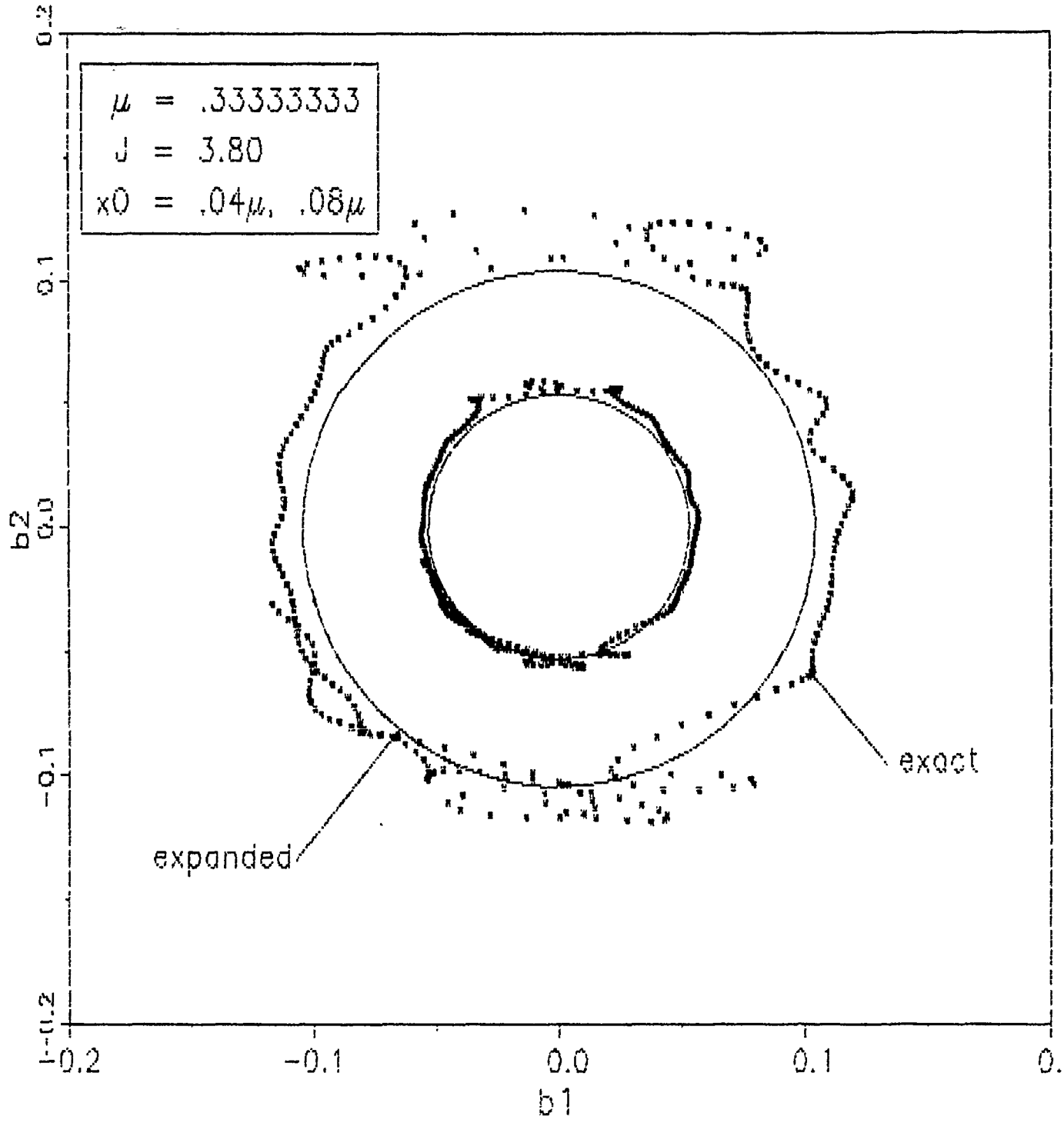


Figure 26. Overlay of Exact and Expanded Nearly-periodic Trajectories in a Transition From Two Exact Integrals of Motion to One (Highly Perturbed System)

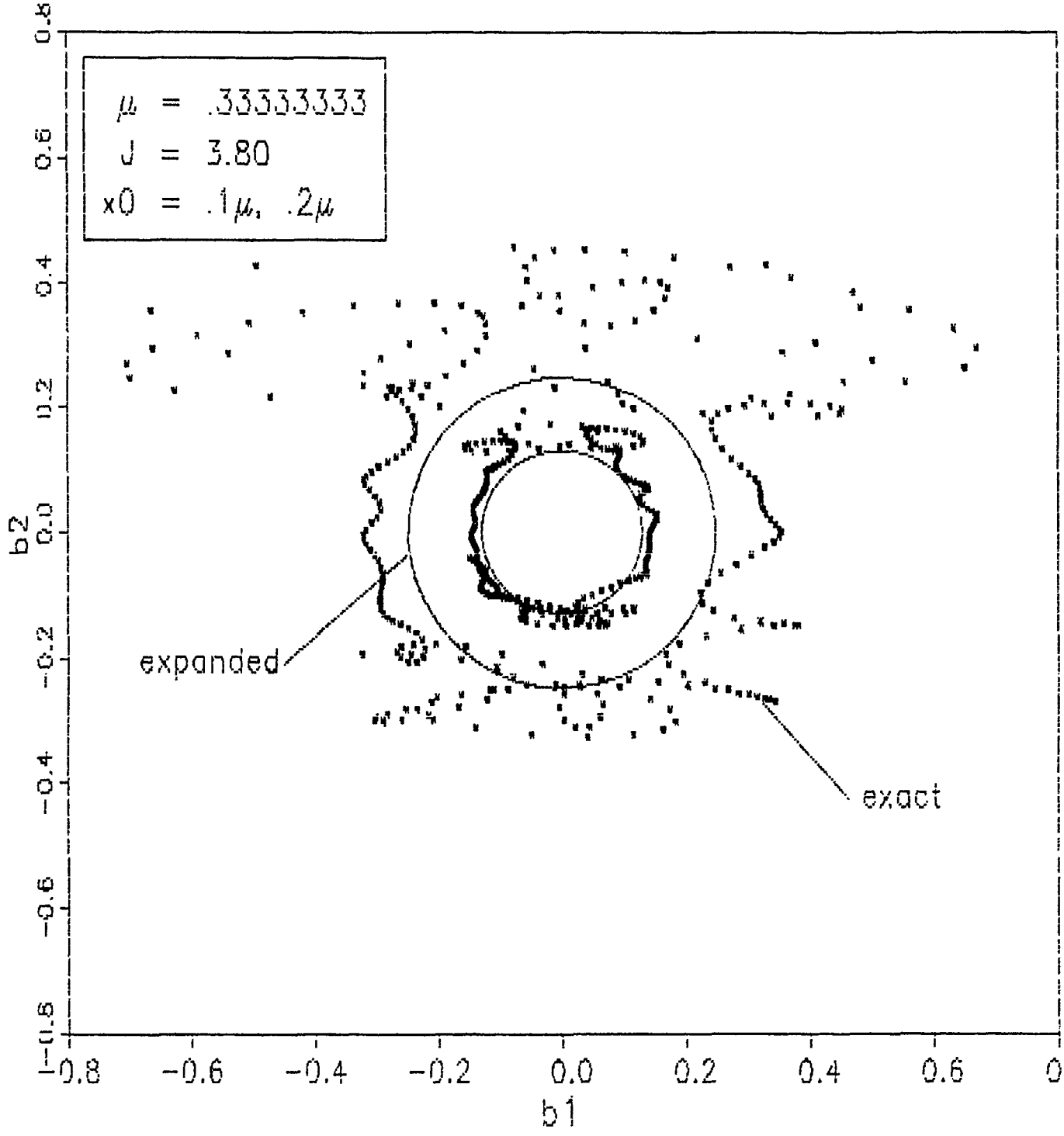


Figure 27. Overlay of Exact and Expanded Nearly-periodic Trajectories in the Absence of a Second Integral of Motion (Highly Perturbed System)

VII. Conclusions and Recommendations

The Modal Transformation and the Limits of the Tangent Space

The transformation into modal variables does indeed map the state vector into a set of two variables and two constants in a small region surrounding the periodic orbit. In the Sun-Jupiter system, the radius of displacement separating the regions where the epoch was or wasn't constant, appears to be about 10% of μ or approximately 1.0E-4. The Hamiltonian constant remained so, for every displacement attempted. This is interpreted to mean that the Hamiltonian surface is relatively flat over a large area, which permits an extended region of alignment between the Hamiltonian surface and the tangent space. Given the emergence of epoch time as a variable, this technique could be applied to two-body systems near a resonance.

In the highly perturbed case, the apparent two-integral region occurred within 1% of μ. While this is a much smaller percentage of μ than in the previous case, the actual size of the region is 3.0E-3. In terms of a ratio, this region is 35 times as large as the region in the Sun-Jupiter system. Although this result was not expected, it makes sense in terms of the location of the center of mass of the three-body system. In the Sun-Jupiter system, the mass center is nearly on the sun, where as the mass center for the highly perturbed case is a third of the distance separating the primaries away from the primary $1-\mu$.

As before, a transition of the epoch constant to the epoch variable was observed with larger displacements from the periodic

trajectory. In this case, however, the Hamiltonian constant began to move as well. Both observations coincided with the transition to chaos. It appears that the Hamiltonian surface and the tangent space skew much more abruptly than before. Unlike the Sun-Jupiter system, the useable portion of the phase space is strictly limited to the region where both integrals exist.

The Expanded Vs. the Exact Equations of Motion

The exact and expanded versions of the nearly-periodic trajectories correlate completely in the presence of both constants of motion. For the Sun-Jupiter case, within 10% of μ, and for the highly perturbed case, within 1% of μ. These limits were determined by simple observation of the figures presented, and are not intended to be exact. In these cases, however, the extra terms in the expanded Hamiltonian equations of motion eq(68), other than the first, are unnecessary. These extra terms are either zero, or very small, and fail to accurately model the transition of the epoch variable. Therefore, if the desired trajectories are strictly limited to the two-integral regions, then the equations of motion could be further reduced to

$$\begin{aligned} \dot{b}_1 &= \omega b_2 \\ \dot{b}_2 &= -\omega b_1 \end{aligned} \tag{69}$$

which are the equations of a harmonic oscillator. Since this system can be solved in closed-form, no integration would be necessary. The programs EXPANDED and FLOQUET/HAMILTONIAN could be eliminated completely.

Conversely, if the nearly-periodic trajectories outside of the two-integral region are to be explored, then more terms must be maintained in the Taylor's expansion of the new Hamiltonian. Eventually, by comparison of the new expanded trajectories against the exact ones, the proper truncation limit may be determined. Once this has been accomplished the perturbation solution may then be constructed.

Either way, a solution for nearly-periodic orbits can be derived in closed-form. The entire system would be modelled as a harmonic oscillator with a forcing term, and no further integration would be necessary. The final task would then be to derive a functional relationship between the system parameters and integrals of motion, and the maximum allowable initial displacement from the periodic orbit.

$$\left(b_1(0), b_2(0)\right)_{max} = f(b_3, b_4, \mu)$$

Appendix A: Code Validation and Error Determination

To ensure that the computer programs used in this study were correct, a large amount of testing and cross checking was required during code development. Wherever possible, hand checks were performed. Unfortunately, given the numerical complexity of most of these programs, few were actually validated that way.

The Surface of Section

It was very easy to determine the accuracy of this program. Assuming that the catalog of SOS plots published by Jefferys was correct, the program was complete when the pictures matched. This step validated the equations of motion for the restricted three-body problem, as well as the surfacing technique.

Periodic Orbit Determination Program

The simple beauty of a periodic orbit is that it returns to it's original conditions after each orbit. The state transition matrix was checked by taking numerical derivatives, and it's eigenvectors using a linear algebra software package. The last, best test was a hand check of the converged state values substituted into the equations of motion.

Floquet/Fourier and Floquet/Hamiltonian

Both of these programs integrated the equations of motion already validated, using the periodic initial conditions, also already checked. Even though the individual outputs were different, the methods used were the same. Periodic state or

Hamiltonian information was extracted at fixed intervals during the integration. The only check available here was to scrutinize the general periodic trends in the data. Upon completion of the integration, the discrete periodic information was converted into Fourier coefficients. These in turn were checked by simply reconstructing the known trajectory from the coefficients.

The Program Exact

Here, the only process yet to be checked was the matrix inversion from the intermediate set of coordinates to the modal ones. This was accomplished via the linear algebra package. The two constants were checked for any periodic modulation that may have been caused by insufficient sampling of the periodic trajectory. In these cases the number of Fourier samples was simply increased.

The Program Expanded

The only test left to perform, was to measure the size and shape of the output compared to the exact case, using identical inputs.

Appendix B: Thesis Software

```
ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
c
c     PROGRAM SECTION
c
c     PURPOSE: Creates a surface of section plot file
c
c     SUBROUTINES: HAMING.F
c                  RHS1.F
c                  H.F
c
ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc

      program sect
c
c     problem commons
c
      common /data/ xmu,xmua
      common /lam/ xlambda(4)
      common /ham/ t,x(20,4),f(20,4),err(20),nn,hh,mode
c
c     variable declarations
c
      implicit double precision (a-h)
      implicit integer (i-n)
      implicit double precision (o-z)
      character*10 filnam1,filnam2

      dimension xlambda(4),x(20,4),f(20,4),err(20),rdotv(4)
c
c     input data
c
      read(*,*) xmu,xmua
      read(*,*) hh,tmax
      npts = dint(tmax/hh)
      read(*,*) xnot,ynot
      read(*,*) xjac,syn
      read(*,*) filnam1
      read(*,*) filnam2
c
c     open output files
c     -- filnam1 is the plotfile
c     -- filnam2 is a general output file
c
      open(2,file=filnam1,status='unknown')
      open(3,file=filnam2,status='unknown')

      write(3,*) 'mu=',xmu,' 1-mu=',xmua
      write(3,*) 'x0=',xnot,'y0=',ynot
      write(3,*) 'jacobi const=',xjac,'timestep=',hh
      write(3,*)
      write(3,*) 'npts value at n*period'

      mode = 0
      nn = 4
      nxt = 0
      t = 0.d0
c
c     get q1,p1,q2,p2 for given x0,y0, and jacobian
c
      q1 = xnot + xmu
      q2 = ynot
      xham = (xmu*xmua-xjac)/2.d0
      r1 = ((q1-xmu)**2.d0 + q2**2.d0)**.5d0
      r2 = ((q1+xmua)**2.d0 + q2**2.d0)**.5d0
      d = xham + xmua/r1 + xmu/r2
      g = q2/(q1-xmu)
      a = g*g + 1.d0
      b = -2.d0*(g*g*xmu + g*q2 + q1)
      c = (g*xmu)**2.d0 + 2.d0*g*q2*xmu - 2.d0*d
      p2 = (-b+syn*(b*b-4.d0*a*c)**.5d0)/(2.d0*a)
      p1 = g*(xmu-p2)
c
c     initial conditions
c
      x(1,1) = q1
      x(2,1) = p1
      x(3,1) = q2
      x(4,1) = p2
c
c     initialize haming
c
```

```
      call haming(nxt)
c
c     turn off second EOM eval
c
      nxt = -nxt
      if(nxt .ne. 0) go to 499
            stop 99
  499 continue
c
c     integration loop
c
      do 500 i = 1,npts
c
c           permute indices
c
            nm3 = nm2
            nm2 = nm1
            nm1 = nxt
c
c           integrate orbit, haming permutes nxt
c
            call haming(nxt)
c
c           calculate r dot v
c
            q1d = f(1,nxt)
            q2d = f(3,nxt)
            rdotv(nxt) = (x(1,nxt)-xmu)*q1d + x(3,nxt)*q2d
c
c           check for peri/apoapse crossing
c
            if (rdotv(nxt)*rdotv(nm1).gt.0.d0) go to 500
c
c           crossing has occured!!!
c           interpolate to crossing time
c
            frac = -rdotv(nxt)/( rdotv(nxt) - rdotv(nm1) )
            q1c = -frac*x(1,nm1) + (1.d0 + frac)*x(1,nxt)
            q2c = -frac*x(3,nm1) + (1.d0 + frac)*x(3,nxt)
            xcross = q1c - xmu
            ycross = q2c
c
c           compute conjugate momenta p1c and p2c for q1c and q2c
c
            r1c = ((q1c-xmu)**2.d0 + q2c**2.d0)**.5d0
            r2c = ((q1c+xmua)**2.d0 + q2c**2.d0)**.5d0
            dd = xham + xmua/r1c + xmu/r2c
            gg = q2c/(q1c-xmu)
            aa = gg*gg + 1.d0
            bb = -2.d0*(gg*gg*xmu + gg*q2c + q1c)
            cc = (gg*xmu)**2.d0 + 2.d0*gg*q2c*xmu - 2.d0*dd
            p2c = (-bb+syn*(bb*bb-4.d0*aa*cc)**.5d0)/(2.d0*aa)
            p1c = gg*(xmu-p2c)

            write(3,*) i,x(1,nxt)
            write (2,*) xcross,ycross

  500 continue

      close(2)
      close(3)
      stop
      end
```

```
ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
c       PROGRAM PERIOD
c
c       PURPOSE: Calculate periodic trajectory via iterative
c                integration technique.  Once found, determines
c                eigenvalues, eigenvectors, and Poincare exponents
c                for periodic trajectory at t=0.
c
c       SUBROUTINES: HAMING.F
c                    RHS1.F
c                    H.F
c
ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc

      program po

      common /lam/ xlambda(4)
      common /data/ xmu,xmua
      common /ham/ t, x(20,4),f(20,4),err(20),nn,hh,mode
c
c     local variables
c
      implicit double precision (a-h)
      implicit integer (i-n)
      implicit double precision (o-z)

      dimension xlambda(4),x(20,4),f(20,4),err(20),cerr(2,1),b(2,2)
      dimension phi(4,4),xxx(10),wk(50),xww(2),rvec(2,16)
      dimension tvec(16,2),alpha(4),tnvec(16,2),xreal(4),ximag(4)
      complex*16 w(4),vec(4,4),ww
      equivalence (ww,xww)
      equivalence (vec,rvec)
      character*10 filnam1,filnam2
c
c        read input data
c
      read(*,*) xmu,xmua
      read(*,*) period,npts
      read(*,*) x0,y0
      read(*,*) tol,maxit
      read(*,*) xjacob
      read(*,*) filnam1
      read(*,*) filnam2
c
c        calculate timestep
c
      hh = period/(dble(npts))
c
c        calculate q1,p1,q2,p2 for given x0,y0, and jacobian
c
      q1 = x0 + xmu
      q2 = y0
      xham = (xmu*xmua-xjacob)/2.d0
      r1 = ((q1-xmu)**2.d0 + q2**2.d0)**.5d0
      r2 = ((q1+xmua)**2.d0 + q2**2.d0)**.5d0
      dd = xham + xmua/r1 + xmu/r2
      gg = q2/(q1-xmu)
      aa = gg*gg + 1.d0
      bb = -2.d0*(gg*gg*xmu + gg*q2 + q1)
      cc = (gg*xmu)**2.d0 + 2.d0*gg*q2*xmu - 2.d0*dd
      p2 = (-bb+(bb*bb-4.d0*aa*cc)**.5d0)/(2.d0*aa)
      p1 = gg*(xmu-p2)
c
c        echo inputs to output file
c
      open(3,file=filnam1,status='unknown')

      write(3,*) 'xmu = ',xmu,'  xmua = ',xmua
      write(3,*) 'orbit period, npts ',period,npts
      write(3,*) 'timestep ',hh
      write(3,*) 'q10 = ',q1,'  p10 = ',p1
      write(3,*) 'q20 = ',q2,'  p20 = ',p2
      write(3,*) 'tol, maxit',tol,maxit
c
c        begin iteration loop
c
      do 1000 iter = 1,maxit
c
c           set up initial state
c
        x(1,1) = q1
        x(2,1) = p1
        x(3,1) = q2
        x(4,1) = p2
c
c           write progress
```

```
c
      write(3,*)
      write(3,*) 'iteration',iter
      write(3,*) 'q1=',q1,' p1=',p1
      write(3,*) 'q2=',q2,' p2=',p2
c
c        initialize phi matrix
c
      do 100 i = 1,4
        do 101 j = 1,4
          ij = 4*i+j
          x(ij,1) = 0.d0
  101   continue
        x(5*i,1) = 1.d0
  100 continue
c
c        initialize integration constants
c
      mode = 1
      nn = 20
      nxt = 0
      t = 0.d0
c
c        initialize haming
c
      call haming(nxt)

      if(nxt .ne. 0) go to 499
        write (*,*) 'failure to initialize'
        stop 99
  499 continue
c
c        integration loop
c
      do 500 i = 1,npts
        call haming(nxt)
  500 continue
c
c        extract error vector
c
      cerr(1,1) = -x(2,nxt)
      cerr(2,1) = -x(3,nxt)
c
c        extract correction matrix
c
      b(1,1) = x(9,nxt)
      b(1,2) = x(12,nxt)
      b(2,1) = x(13,nxt)
      b(2,2) = x(16,nxt)
c
c        calculate state corrections
c
      call leqt2f(b,1,2,2,cerr,idig,xxx,ier)
c
c        add in corrections
c
      q1 = q1 + cerr(1,1)
      p2 = p2 + cerr(2,1)
c
c        check for convergence
c
      iend = 0
      if(dabs(cerr(1,1)) .gt. tol) iend = 1
      if(dabs(cerr(2,1)) .gt. tol) iend = 1
      if(iend .eq. 0) go to 2000

 1000 continue

c
c       maximum iterations exceeded without convergence
c
     write (*,*) 'Iteration Limit Exceeded'
     stop
 2000 continue
c
c     converged processing
c
     write(3,*)
     write(3,*) 'program converged in',iter,'iterations'
     write(3,*)
     write(3,*) 'converged state values'
     write(3,*) 'q1=',x(1,nxt),'p1=',x(2,nxt)
     write(3,*) 'q2=',x(3,nxt),'p2=',x(4,nxt)
     write(3,*)
     write(3,*) 'surface of section coordinates'
     write(3,*) 'x =',x(1,nxt)-xmu,'y =',x(3,nxt)
```

```
      write(3,*)
c
c     compute hamiltonian/jacobian
c
      q1 = x(1,nxt)
      p1 = x(2,nxt)
      q2 = x(3,nxt)
      p2 = x(4,nxt)
      r1 = dsqrt((q1-xmu)**2.d0 + q2**2.d0)
      r2 = dsqrt((q1+xmua)**2.d0 + q2**2.d0)
      xham = .5d0*(p1*p1+p2*p2) + p1*q2 - p2*q1 - xmua/r1 - xmu/r2
      xjac = xmu*xmua - 2.d0*xham

      write(3,*) 'ham =',xham,'jac =',xjac
c
c     extract phi
c
      do 2005 i = 1,4
        do 2005 j = 1,4
          phi(i,j) = x(4*i+j,nxt)
 2005 continue
c
c     compute eigen values and vectors of phi
c
      call eigrf(phi,4,4,2,w,vec,4,wk,ier)
c
c        transpose rvec, store as tvec
c
      do 19 i=1,16
        ii = (i/4.1)+1
          alpha(ii) = 0.d0
            do 19 j=1,2
              tvec(i,j) = rvec(j,i)
   19 continue
c
c        normalize eigenvector matrix
c
      do 21 i=1,16
        ii = (i/4.1)+1
          do 21 j=1,2
             alpha(ii) = alpha(ii) + tvec(i,j)**2.d0
   21 continue

      do 23 i=1,16
        ii = (i/4.1)+1
          do 23 j=1,2
            tnvec(i,j) = tvec(i,j)/dsqrt(alpha(ii))
   23 continue

      write(3,*)
      write(3,*) 'normalized eigenvectors of phi, by column'
      do 24 i=1,16
          write(3,7) tnvec(i,1),tnvec(i,2)
   24 continue
    7 format(1x,2(f20.13,1x))
c
c     compute Poincare exponents
c
      write(3,*)
      write(3,*) 'Poincare exponents'
      do 2100 i = 1,4
        ww = w(i)
c           complex log of eigenvalue over period
        xreal(i) = dlog(dsqrt(xww(1)*xww(1) + xww(2)*xww(2)))
     *               / period
        ximag(i) = datan2( xww(2), xww(1) ) / period
        write(3,5) xreal(i),ximag(i)
 2100 continue
    5 format(3x,2(e20.13,1x))
c
c        create input file for flo.f and fho.f
c
      open(2,file=filnam2,status='unknown')

      write(2,*) xmu,' ',xmua
      write(2,*) period,' ',npts
      write(2,*) x(1,nxt),' ',x(4,nxt)
      write(2,*)
c
c        write normalized eigenvector parts by column
c        -- if complex first leave alone
c        -- if real first, switch order
c
      jt = 0

      if ((tnvec(1,2).eq.0.d0).and.(tnvec(2,2).eq.0.d0)) jt=8
```

```
      do 25 j=1,2
        do 25 i=1+jt,4+jt
          write(2,*) tnvec(i,j)
 25   continue

      do 26 i=9-jt,16-jt
        write(2,*) tnvec(i,1)
 26   continue
c
c        write poincare exponents for j matrix
c          -- if complex first leave alone
c          -- if real first, then switch
c
      if (ximag(1).eq.0.d0) then

        write(2,*)
        write(2,*) xreal(3)
        write(2,*) ximag(4)
        write(2,*) 0.d0
        write(2,*) 0.d0
        write(2,*) ximag(3)
        write(2,*) xreal(4)
        write(2,*) 0.d0
        write(2,*) 0.d0
        write(2,*) 0.d0
        write(2,*) 0.d0
        write(2,*) xreal(1)
        write(2,*) ximag(2)
        write(2,*) 0.d0
        write(2,*) 0.d0
        write(2,*) ximag(1)
        write(2,*) xreal(2)
      else
        write(2,*)
        write(2,*) xreal(1)
        write(2,*) ximag(2)
        write(2,*) 0.d0
        write(2,*) 0.d0
        write(2,*) ximag(1)
        write(2,*) xreal(2)
        write(2,*) 0.d0
        write(2,*) 0.d0
        write(2,*) 0.d0
        write(2,*) 0.d0
        write(2,*) xreal(3)
        write(2,*) ximag(4)
        write(2,*) 0.d0
        write(2,*) 0.d0
        write(2,*) ximag(3)
        write(2,*) xreal(4)
      endif

      close(2)
      close(3)

      stop
      end
```

```
cccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
c     PROGRAM FLOQUET/FOURIER
c
c     PURPOSE: Propagates the state and eigenvectors of a periodic
c              orbit as determined by program period.  Every value
c              during the integration is saved, and then converted
c              into a fourier series.
c
c     SUBROUTINES: HAMING.F
c                  RHS2.F
c                  H.F
c                  FOURIER.F
c
Ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc

      program fl

      common /lam/ xlambda(4)
      common /data/ xmu,xmua,xj(4,4)
      common /ham/ t,x(20,4),f(20,4),err(20),nn,hh,mode
c
c     local variables
c
      implicit double precision (a-h)
      implicit integer (i-n)
      implicit double precision (o-z)
      dimension xlambda(4),x(20,4),f(20,4),err(20),temp(100),ck(51)
      dimension xic(2),xj(4,4),x0(16),s(4,100),v(4,4,100),sk(51)
c
c     read input data
c
      read (*,*) xmu,xmua
      read (*,*) period,npts
      read (*,*) xic(1),xic(2)

      do 10 i=1,4
      do 10 j=1,4
        ii = (j-1)*4 + i
        read (*,*) x0(ii)
  10  continue

      do 20 i=1,4
        do 20 j=1,4
          read (*,*) xj(j,i)
  20  continue

      hh = period/(dble(npts))

      write (*,*) 'xmu = ',xmu,'  xmua = ',xmua
      write (*,*) 'orbit period, npts ',period,npts
      write (*,*) 'timestep ',hh
      write (*,*) 'initial conditions (q1, p1=0, q2=0, p2)'
      write (*,*) 'q1 = ',xic(1),'  p2 = ',xic(2)
      write (*,*)

      write (*,*) 'f(0)'
        do 30 i=1,13,4
          write (*,1) x0(i),x0(i+1),x0(i+2),x0(i+3)
  30    continue
  1   format(1x,f18.10,f18.10,f18.10,f18.10)

      write (*,*)
      write (*,*) 'xj(i,j)'
        do 40 i=1,4
          write (*,1) xj(i,1),xj(i,2),xj(i,3),xj(i,4)
  40    continue
c
c           set up initial state
c
            x(1,1) = xic(1)
            x(2,1) = 0.d0
            x(3,1) = 0.d0
            x(4,1) = xic(2)
c
c           initialize f(0) matrix
c
            do 160 i = 1,16
              x(i+4,1) = x0(i)
  160       continue

            mode = 1
            nn = 20
            nxt = 0
            t = 0.d0
c
c           initialize haming
```

```
c
          call haming(nxt)

          if(nxt .ne. 0) go to 499
                write (*,*) 'failure to initialize'
               write (*,*) f(1,1),f(2,1)
               write (*,*) f(3,1),f(4,1)
               stop 99
  499     continue
c
c         integration loop
c
         do 500 i = 1,100
            do 501 j = 1,4
              s(j,i) = x(j,nxt)
                do 501 k = 1,4
                  v(j,k,i) = x(4*j+k,nxt)
  501               continue
            do 502 m = 1,10
              call haming(nxt)
  502           continue
  500     continue
c
c       open output file
c
      open(2,file='coef.fou',status='unknown')
c
c       copy eig values/vectors and feed to fourier
c
       do 515 j=1,4
         do 510 i=1,100
           temp(i) = s(j,i)
  510        continue
         call fourier(temp,ck,sk,50)
             do 520 k=1,50
               write(2,*) ck(k),sk(k)
  520          continue
  515  continue

       do 525 i=1,4
         do 525 j=1,4
           do 530 k=1,100
             temp(k) = v(j,i,k)
  530          continue
           call fourier(temp,ck,sk,50)
               do 535 m=1,50
                 write(2,*) ck(m),sk(m)
  535              continue
  525  continue
c
c     final state conditions
c
      write (*,*)
      write(*,*) 'state at tf'
      write(*,*) '   q1=',x(1,nxt),'   p1=',x(2,nxt)
      write(*,*) '   q2=',x(3,nxt),'   p2=',x(4,nxt)

      write (*,*)
      write (*,*) 'f(t)'
      do 600 i=5,17,4
        write (*,1) x(i,nxt),x(i+1,nxt),x(i+2,nxt),x(i+3,nxt)
  600 continue

      close(2)
      stop
      end
```

```
ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
c     PROGRAM FLOQUET/HAMILTONIAN
c
c     PURPOSE: Calculate period coefficients needed in the new
c              expanded hamiltonian, from the third order
c              hamiltonian of the periodic trajectory.  Compute
c              after each integration step, and convert the result
c              into a fourier series.
c
c     SUBROUTINES: HAMING.F
c                  RHS2.F
c                  H.F
c                  FOURIER.F
c
Ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc

      program fh

      common /lam/ xlambda(4)
      common /data/ xmu,xmua,xj(4,4)
      common /ham/ t,x(20,4),f(20,4),err(20),nn,hh,mode
c
c     local variables
c
      implicit double precision (a-h)
      implicit integer (i-n)
      implicit double precision (o-z)
      dimension xlambda(4),x(20,4),f(20,4),err(20),xh3(4,4,4)
      dimension xic(2),xj(4,4),x0(16),xx(4),v1(4),v2(4)
      dimension tc(8),c(4,100),ck(100),sk(100),temp(100)
c
c     read input data
c
           read (*,*) xmu,xmua
           read (*,*) period,npts
           read (*,*) xic(1),xic(2)
c
c     transpose col to row to fit x(20)
c
           do 10 i=1,4
             do 10 j=1,4
               ii = (j-1)*4 + i
               read (*,*) x0(ii)
  10       continue

           do 20 i=1,4
             do 20 j=1,4
               read (*,*) xj(j,i)
  20       continue

           hh = period/(dble(npts))
c
c      output inputs
c
           write (*,*) 'xmu = ',xmu,'  xmua = ',xmua
           write (*,*) 'orbit period, npts ',period,npts
           write (*,*) 'timestep ',hh
           write (*,*) 'initial conditions (q1, p1=0, q2=0, p2)'
           write (*,*) 'q1 = ',xic(1),'  p2 = ',xic(2)
           write (*,*)

           write (*,*) 'f(0)'
           do 30 i=1,13,4
             write (*,1) x0(i),x0(i+1),x0(i+2),x0(i+3)
  30       continue
  1        format(1x,4(f18.10))

           write (*,*)
           write (*,*) 'xj(i,j)'
           do 40 i=1,4
             write (*,1) xj(i,1),xj(i,2),xj(i,3),xj(i,4)
  40       continue
c
c     set up initial state
c
           x(1,1) = xic(1)
           x(2,1) = 0.d0
           x(3,1) = 0.d0
           x(4,1) = xic(2)
c
c     initialize f(0) matrix
c
           do 160 i = 1,16
             x(i+4,1) = x0(i)
  160      continue
```

```
      mode = 1
      nn = 20
      nxt = 0
      t = 0.d0
c
c     initialize haming
c
      call haming(nxt)

      if(nxt .ne. 0) go to 499
        write (*,*) 'failure to initialize'
        write (*,*) f(1,1),f(2,1)
        write (*,*) f(3,1),f(4,1)
              stop 99
  499 continue
c
c     begin integration loop
c
      do 500 i = 1,100
        do 510 j = 1,4
          xx(j) = x(j,nxt)
          v1(j) = x(4*j+1,nxt)
          v2(j) = x(4*j+2,nxt)
  510   continue
c
c       compute third order hamiltonian tensor
c
        do 520 j = 1,4
        do 520 k = 1,4
        do 520 m = 1,4
          xh3(j,k,m) = h(xx,3,j,k,m,0,0)
  520   continue

        do 525 j=1,8
          tc(j) = 0.d0
  525   continue
c
c       compute periodic coefficients
c
        do 530 j=1,4
        do 530 k=1,4
        do 530 m=1,4
          tc(1) = tc(1) + xh3(j,k,m) * v1(j) * v1(k) * v1(m)
          tc(2) = tc(2) + xh3(j,k,m) * v1(j) * v2(k) * v1(m)
          tc(3) = tc(3) + xh3(j,k,m) * v2(j) * v1(k) * v1(m)
          tc(4) = tc(4) + xh3(j,k,m) * v1(j) * v1(k) * v2(m)
          tc(5) = tc(5) + xh3(j,k,m) * v2(j) * v2(k) * v1(m)
          tc(6) = tc(6) + xh3(j,k,m) * v1(j) * v2(k) * v2(m)
          tc(7) = tc(7) + xh3(j,k,m) * v2(j) * v1(k) * v2(m)
          tc(8) = tc(8) + xh3(j,k,m) * v2(j) * v2(k) * v2(m)
  530   continue

        c(1,i) = tc(1)/6.d0
        c(2,i) = (tc(2)+tc(3)+tc(4))/6.d0
        c(3,i) = (tc(5)+tc(6)+tc(7))/6.d0
        c(4,i) = tc(8)/6.d0

        do 550 m = 1,10
          call haming(nxt)
  550   continue
  500 continue
c
c     compute fourier coefficients from periodic ones
c
      open(2,file='coef.ham',status='unknown')

      do 570 i=1,4
        do 580 j=1,100
          temp(j) = c(i,j)
  580   continue

        call fourier(temp,ck,sk,50)

        do 590 k=1,50
          write(2,*) ck(k),sk(k)
  590   continue
        write(2,*)
  570 continue
c
c     final state conditions
c
      write (*,*)
      write(*,*) 'state at tf'
      write(*,*) '   q1=',x(1,nxt),'   p1=',x(2,nxt)
      write(*,*) '   q2=',x(3,nxt),'   p2=',x(4,nxt)
      write (*,*)
```

```
      write (*,*) 'f(t)'
      do 600 i=5,17,4
        write (*,1) x(i,nxt),x(i+1,nxt),x(i+2,nxt),x(i+3,nxt)
600   continue

    close(2)
    stop
    end
```

```
ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
c     PROGRAM EXACT
c
c     PURPOSE: integrates a nearly periodic trajectory, subtracts
c              the periodic reference, and transforms the result
c              into modal variables, and creates a plotfile
c              b1 vs b2.
c
c     SUBROUTINES: HAMING.F
c                  RHS1.F
c                  H.F
c
Ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc

      program check
c
c     problem commons
c
      common /data/ xmu,xmua
      common /lam/ xlambda(4)
      common /ham/ t,x(20,4),f(20,4),err(20),nn,hh,mode
c
c     variable declarations
c
      implicit double precision (a-h)
      implicit integer (i-n)
      implicit double precision (o-z)
      character*10 filnam1,filnam2,filnam3

      dimension xlambda(4),x(20,4),f(20,4),err(20)
      dimension sinn(50),coss(50),ck(20,50),sk(20,50),cf(20)
      dimension cc(4,4),xx(4),dx(4),xxx(50)
c
c     input data
c
         read(*,*) xmu,xmua
         read(*,*) period,npts
         hh = period/(dble(npts))
         read(*,*) xnot,ynot
         read(*,*) xjac,syn
         read(*,*) w,trip
         read(*,*) filnam1
         read(*,*) filnam2
         read(*,*) filnam3


         do 100 i=1,20
           do 100 j=1,50
              read(*,*) ck(i,j),sk(i,j)
  100    continue

         mode = 0
         nn = 4
         nxt = 0
         t = 0.d0

         pi = dacos(-1.d0)
         w0 = 2.d0*pi/period
c
c     get q1,p1,q2,p2 for given x0,y0, and jacobian
c
         q1 = xnot + xmu
         q2 = ynot
         xham = (xmu*xmua-xjac)/2.d0
         r1 = ((q1-xmu)**2.d0 + q2**2.d0)**.5d0
         r2 = ((q1+xmua)**2.d0 + q2**2.d0)**.5d0
         d = xham + xmua/r1 + xmu/r2
         g = q2/(q1-xmu)
         a = g*g + 1.d0
         b = -2.d0*(g*g*xmu + g*q2 + q1)
         c = (g*xmu)**2.d0 + 2.d0*g*q2*xmu - 2.d0*d
         p2 = (-b+syn*(b*b-4.d0*a*c)**.5d0)/(2.d0*a)
         p1 = g*(xmu-p2)

         write(*,*) 'initial conditions (Jefferys)'
         write(*,*) 'x0=',xnot,' y0=',ynot
         write(*,*)
         write(*,*) 'initial conditions (Szebehely)'
         write(*,*) 'q1=',q1,' p1=',p1
         write(*,*) 'q2=',q2,' p2=',p2
         write(*,*)
         write(*,*) 'jacobian=',xjac
         write(*,*) 'period=',period
         write(*,*)
         write(*,*) 'initial conditions (Modal)'
c
```

```
c     initial conditions
c
         x(1,1) = q1
         x(2,1) = p1
         x(3,1) = q2
         x(4,1) = p2
c
c     initialize haming
c
         call haming(nxt)
c
c     turn off second EOM eval
c
         nxt = -nxt
         if(nxt .ne. 0) go to 499
            stop 99
  499    continue
c
c     open output files
c
         open(2,file=filnam1,status='unknown')
         open(3,file=filnam2,status='unknown')
         open(4,file=filnam3,status='unknown')
c
c     integration loop
c
         do 500 i = 0,npts*trip

            if (mod(i,20).eq.0) then

c
c     compute sin(n*theta), cos(n*theta), n=1 to 50
c
               coss(1) = dcos(w0*t)
               sinn(1) = dsin(w0*t)
               coss(2) = 2.d0*coss(1)*coss(1) - 1.d0
               sinn(2) = 2.d0*sinn(1)*coss(1)

               do 200 j=3,50
                 coss(j) = 2.d0*coss(j-1)*coss(1) - coss(j-2)
                 sinn(j) = 2.d0*sinn(j-1)*coss(1) - sinn(j-2)
  200          continue
c
c      reassemble periodic traj and eigenvector matrix
c
               do 300 k=1,20
                cf(k) = ck(k,1)
                 do 300 j=1,49
                  cf(k) = cf(k) + ck(k,j+1)*coss(j)
     *                    + sk(k,j+1)*sinn(j)
  300          continue

c              write(*,*)
c              do 301 j = 1,20
c              write(*,*) cf(j)
c 301          continue

c
c      compute generalized eigenvector (grad of hamiltonian)
c
               do 320 j=1,4
                 xx(j) = cf(j)
  320          continue

               temp = 0.d0
               do 330 j=1,4
                 dx(j) = x(j,iabs(nxt)) - cf(j)
                 cf(16+j) = h(xx,1,j,0,0,0,0)
                 temp = temp + cf(16+j)*cf(16+j)
  330          continue

c              write(*,*)
c              do 331 j=1,4
c              write(*,*) cf(16+j)
c 331          continue


               do 335 j=1,4
                 cf(16+j) = cf(16+j)/(dsqrt(temp))
  335          continue

c              write(*,*)
c              do 336 j=1,20
c              write(*,*) cf(j)
c 336          continue
c
```

```
c       place eigenvectors in 4x4 matrix (for inversion)
c
                do 340 j=1,4
                do 340 k=1,4
                  jj = 4*j+k
                  cc(k,j) = cf(jj)
  340           continue

c
c       invert delta x = eigenvector matrix * b
c
                idig = 0

                call leqt2f(cc,1,4,4,dx,idig,xxx,ier)

                if (i.eq.0.d0) then
                  write(*,*) 'b10=',dx(1),' b20=',dx(2)
                endif

                write(2,*) dx(1),dx(2)
                write(3,*) t,dx(3)
                write(4,*) t,dx(4)

              endif

              call haming(nxt)

  500     continue

      close(2)
      stop
      end
```

```
ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
c     PROGRAM EXPANDED
c
c     PURPOSE: Using the periodic coefficients made by the program
c              floquet/hamiltonian, the eom for the truncated
c              hamiltonian case are integrated.  A plotfile
c              matching b1 vs b2 is created.
c
c     SUBROUTINES: HAMING.F
c                  RHS3.F
c
ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc

      program k3

      common /data/ w0,w,ck(4,50),sk(4,50)
      common /ham/ t,x(20,4),f(20,4),err(20),nn,hh,mode
c
c     local variables
c
      implicit double precision (a-h)
      implicit integer (i-n)
      implicit double precision (o-z)
      character*10 filnam

      dimension x(20,4),f(20,4),err(20)
      dimension ck(4,50),sk(4,50)
c
c     read input data
c
           read (*,*) period,npts
           read (*,*) b10,b20
           read (*,*) w,trip
           read(*,*) filnam

c
c     read fourier coefficients
c
           do 20 i=1,4
             do 20 j=1,50
               read (*,*) ck(i,j),sk(i,j)
  20       continue

           hh = period/(dble(npts))
c
c      output inputs
c
           write (*,*) 'orbit period, npts ',period,npts
           write (*,*) 'timestep ',hh
           write (*,*) 'initial conditions (modal)'
           write (*,*) 'b10=',b10,'  b20=',b20
c
c     set up initial state
c
           x(1,1) = b10
           x(2,1) = b20

           mode = 0
           nn = 4
           nxt = 0
           t = 0.d0

           pi = dacos(-1.d0)
           w0 = 2.d0*pi/period
c
c     initialize haming
c
           call haming(nxt)

           if(nxt .ne. 0) go to 499
             write (*,*) 'failure to initialize'
             write (*,*) f(1,1),f(2,1)
             write (*,*) f(3,1),f(4,1)
                    stop 99
  499      continue
c
c     begin integration loop
c
           open(2,file=filnam,status='unknown')

           do 500 i = 1,npts*trip
              call haming(nxt)
              if (mod(i,100).ne.0) go to 500
              write(2,*) x(1,nxt),x(2,nxt)
  500      continue
```

```
close(2)
stop
end
```

```
ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
c
c     SUBROUTINE HAMING
c
c     PURPOSE: Haming is an ordinary differential equations
c              integrator.  It is a fourth order
c              predictor-corrector algorithm that carries the last
c              four values of the state vector, extrapolates
c              them to obtain the next value (the prediction part),
c              and then corrects the extrapolated value to find a
c              new value for the state vector.  Nxt specifies which
c              of the 4 values of the state vector is the "next"
c              one.  Nxt is updated by haming automatically, and is
c              zero on the first call.  The user supplies an
c              external routine rhs(nxt), which evaluates the
c              equations of motion
c
ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc

      subroutine haming(nxt)

      common /ham/ x,y(20,4),f(20,4),errest(20),n,h,mode
      implicit double precision (a-z)
      dimension y(20,4),f(20,4),errest(20)
      integer i,l,nxt,n,npl,nml,nm2,npo,isw,jsw,mode
c
c     x is the independent variable ( time )
c     y(6,4) is the state vector- 4 copies of it, with nxt
c     pointing at the next one
c     f(6,4) are the equations of motion, again four copies
c     a call to rhs(nxt) updates an entry in f
c     errest is an estimate of the truncation error - normally not
c     used
c     n is the number of equations being integrated - 6 or 42 here
c     h is the time step
c     mode is 0 for just EOM, 1 for both EOM and EOV
c
      tol = 0.0000000001

      if(nxt) 190,10,200

   10 xo = x
      hh = h/2.0d+00
      call rhs(1)
      do 40 l = 2,4
      x = x + hh
      do 20 i = 1,n
   20 y(i,l) = y(i,l-1) + hh*f(i,l-1)
      call rhs(l)
      x = x + hh
      do 30 i = 1,n
   30 y(i,l) = y(i,l-1) + h*f(i,l)
   40 call rhs(l)
      jsw = -10
   50 isw = 1
      do 120 i = 1,n
      hh = y(i,1) + h*( 9.0d+00*f(i,1) + 19.0d+00*f(i,2)
     1     - 5.0d+00*f(i,3) + f(i,4) ) / 24.0d+00
      if( dabs( hh - y(i,2)) .lt. tol ) go to 70
      isw = 0
   70 y(i,2) = hh
      hh = y(i,1) + h*( f(i,1) + 4.0d+00*f(i,2) + f(i,3))/3.0d+00
      if( dabs( hh-y(i,3)) .lt. tol ) go to 90
      isw = 0
   90 y(i,3) = hh
      hh = y(i,1) + h*( 3.0d+00*f(i,1) + 9.0d+00*f(i,2) +
     1     9.0d+00*f(i,3) + 3.0d+00*f(i,4) ) / 8.0d+00
      if( dabs(hh-y(i,4)) .lt. tol ) go to 110
      isw = 0
  110 y(i,4) = hh
  120 continue
      x = xo
      do 130 l = 2,4
      x = x + h
  130 call rhs(l)
      if(isw) 140,140,150
  140 jsw = jsw + 1
      if(jsw) 50,280,280
  150 x = xo
      isw = 1
      jsw = 1
      do 160 i = 1,n
  160 errest(i) = 0.0
      nxt = 1
      go to 280
  190 jsw = 2
```

```
      nxt = iabs(nxt)
c
c     this is hamings normal propagation loop -
c
  200 x = x + h
      npl = mod(nxt,4) + 1
      go to (210,230),isw
c     permute the index nxt modulo 4
  210 go to (270,270,270,220),nxt
  220 isw = 2
  230 nm2 = mod(npl,4) + 1
      nm1 = mod(nm2,4) + 1
      npo = mod(nm1,4) + 1
c
c     this is the predictor part
c
      do 240 i = 1,n
      f(i,nm2) = y(i,npl) + 4.0d+00*h*( 2.0d+00*f(i,npo) - f(i,nm1)
     1     + 2.0d+00*f(i,nm2) ) / 3.0d+00
  240 y(i,npl) = f(i,nm2) - 0.925619835*errest(i)

c
c     now the corrector - fix up the extrapolated state
c     based on the better value of the equations of motion
c
      call rhs(npl)
      do 250 i = 1,n
      y(i,npl) = ( 9.0d+00*y(i,npo) - y(i,nm2) + 3.0d+00*h*( f(i,npl)
     1     + 2.0d+00*f(i,npo) - f(i,nm1) ) ) / 8.0d+00
      errest(i) = f(i,nm2) - y(i,npl)
  250 y(i,npl) = y(i,npl) + 0.0743801653 * errest(i)
      go to (260,270),jsw
  260 call rhs(npl)
  270 nxt = npl
  280 return
      end
```

```
ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
c
c     SUBROUTINE RHS1
c
c     PURPOSE: Computes the right-hand side of the equations of
c              motion and variation, in the restricted three-body
c              problem.
c
ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc

      subroutine rhs(k)
c
c     canonical EOM and EOV, 4th order system
c
      common /lam/ xlambda(4)
      common /data/ xmu, xmua
      common /ham/ t, x(20,4),f(20,4),err(20),nn,hh,mode
c
      double precision t,x,f,err,hh
      double precision xlambda,xmu,xmua
      double precision h,xx(4),z(4,4),grdh(4,4),temp(4,4)
c
      data z/ 0.d0,  -1.d0,  0.d0,  0.d0,
     1        1.d0,   0.d0,  0.d0,  0.d0,
     2        0.d0,   0.d0,  0.d0, -1.d0,
     3        0.d0,   0.d0,  1.d0,  0.d0/
c
c     extract state
c
      do 10 i = 1,4
         xx(i) = x(i,k)
   10 continue
c
c     equations of motion
c
      f(1,k) = h(xx,1,2,0,0,0,0)
      f(2,k) = -h(xx,1,1,0,0,0,0)
      f(3,k) = h(xx,1,4,0,0,0,0)
      f(4,k) = -h(xx,1,3,0,0,0,0)
c
      if(mode .eq. 0) return
c
c    calculate order 2 gradient matrix
c
      do 20 i = 1,4
           do 20 j = 1,4
                grdh(i,j) = h(xx,2,i,j,0,0,0)
   20 continue
c
c     matrix mpy by z
c
      do 30 i = 1,4
           do 30 ii = 1,4
                ij = 4*i+ii
                temp(i,ii) = 0.d0
                do 30 j = 1,4
                  temp(i,ii) = temp(i,ii) + z(i,j)*grdh(j,ii)
   30 continue
c
c     calculate A phi
c
      do 35 i = 1,4
           do 35 ii = 1,4
                ij = 4*i+ii
                f(ij,k) = 0.d0
                do 35 j = 1,4
                     f(ij,k) = f(ij,k) + temp(i,j)*x(4*j+ii,k)
   35 continue
c
c     check if propagating eigenvectors or phi
c
      if(mode .eq. 1) return
c
c     get eigenvector eom
c
      do 40 j = 2,4,2
           do 40 ii = 1,4
                ij = 4*(j-1) + II
                f(ij,k) = f(ij,k) + xlambda(j)*x(ij+4,k)
                f(ij+4,k) = f(ij+4,k) - xlambda(j)*x(ij,k)
   40 continue
      return
      end
```

```
ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
c     SUBROUTINE RHS2
c
c     PURPOSE: Creates right-hand side of the eom for the
c              restricted three-body problem, and for the
c              eigenvector equation.
c
ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc

      subroutine rhs(k)
c
c     canonical EOM and EOV, 4th order system
c
      common /lam/ xlambda(4)
      common /data/ xmu,xmua,xj(4,4)
      common /ham/ t,x(20,4),f(20,4),err(20),nn,hh,mode

      double precision x(20,4),f(20,4),err(20),f1(20,4),f2(20,4)
      double precision xlambda,xmu,xmua,t,hh
      double precision h,xx(4),z(4,4),grdh(4,4),temp(4,4),xj(4,4)

      data z/ 0.d0,  -1.d0,  0.d0,  0.d0,
     1        1.d0,   0.d0,  0.d0,  0.d0,
     2        0.d0,   0.d0,  0.d0, -1.d0,
     3        0.d0,   0.d0,  1.d0,  0.d0/
c
c     extract state
c
      do 10 i = 1,4
         xx(i) = x(i,k)
   10 continue
c
c     equations of motion
c
      f(1,k) = h(xx,1,2,0,0,0,0)
      f(2,k) = -h(xx,1,1,0,0,0,0)
      f(3,k) = h(xx,1,4,0,0,0,0)
      f(4,k) = -h(xx,1,3,0,0,0,0)
c
      if(mode .eq. 0) return
c
c    calculate order 2 gradient matrix
c
      do 20 i = 1,4
           do 20 j = 1,4
                grdh(i,j) = h(xx,2,i,j,0,0,0)
   20 continue
c
c     matrix mpy by z
c
      do 30 i = 1,4
           do 30 ii = 1,4
                ij = 4*i+ii
                temp(i,ii) = 0.d0
                do 30 j = 1,4
                  temp(i,ii) = temp(i,ii) + z(i,j)*grdh(j,ii)
   30 continue
c
c     calculate A phi and phi J
c
      do 35 i = 1,4
           do 35 ii = 1,4
                ij = 4*i+ii
                f1(ij,k) = 0.d0
                f2(ij,k) = 0.d0
                do 35 j = 1,4
                     f2(ij,k) = f2(ij,k) + x(4*i+j,k)*xj(j,ii)
                     f1(ij,k) = f1(ij,k) + temp(i,j)*x(4*j+ii,k)
   35 continue

      do 36 i = 5,20
           f(i,k) = f1(i,k) - f2(i,k)
   36 continue

      return
      end
```

```
ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
c     SUBROUTINE RHS3
c
c     PURPOSE: Calculate rhs for nearly-periodic eom, using
c              expanded hamiltonian.
c
ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc

      subroutine rhs(k)
c
c     canonical EOM and EOV, 4th order system
c
      common /data/ w0,w,ck(4,50),sk(4,50)
      common /ham/ t,x(20,4),f(20,4),err(20),nn,hh,mode

      double precision t,x(20,4),f(20,4),err(20),hh,sinn(50)
      double precision ck(4,50),sk(4,50),c(4),coss(50),w0,w
      double precision b1,b2

c
c        generate sin(1 to 50 * w0) and cos(1 to 50 * w0)
c
         coss(1) = dcos(w0*t)
         coss(2) = 2.d0*coss(1)*coss(1) - 1.d0
         sinn(1) = dsin(w0*t)
         sinn(2) = 2.d0*sinn(1)*coss(1)

         do 100 i=3,50
           coss(i) = 2.d0*coss(i-1)*coss(1) - coss(i-2)
           sinn(i) = 2.d0*sinn(i-1)*coss(1) - sinn(i-2)
  100    continue

c
c        reconstruct periodic function from coefficients
c
         do 200 i=1,4
           c(i) = ck(i,1)
             do 200 j=1,49
               c(i) = c(i) + ck(i,j+1)*coss(j) + sk(i,j+1)*sinn(j)
  200    continue

         b1 = x(1,k)
         b2 = x(2,k)
c
c        calculate b1 dot and b2 dot
c
         f(1,k) = w*b2 - b1*b1*c(2) +
     *              2.d0*b1*b2*c(3) - 3.d0*b2*b2*c(4)
         f(2,k) = -w*b1 - b2*b2*c(3) +
     *              2.d0*b1*b2*c(2) - 3.d0*b1*b1*c(1)

         return
         end
```

```
ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
c
c     FUNCTION H
c
c     PURPOSE: Computes desired order tensor of the restricted
c              three-body hamiltonian.
c
ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc

      function h(x,iord,i,j,k,l,m)
c
c     state vector x = ( q1, p1, q2, p2 )
c
      common /data/ xmu,xmua
      implicit double precision (a-z)
      dimension x(4)
      integer iord,jord,i,j,k
c
c     preliminaries
c
      qa = x(1) - xmu
      qb = x(1) + xmua
      r1 = dsqrt(qa*qa + x(3)*x(3))
      r2 = dsqrt(qb*qb + x(3)*x(3))
c
c     branch on order
c
      jord = iord + 1
      go to (1, 1000, 2000, 3000),jord
c
c    **         Order  Zero            **
c
    1 continue
      h = 0.5d0*(x(2)*x(2) + x(4)*x(4)) + x(3)*x(2) - x(1)*x(4)
     1     - xmua/r1  -  xmu/r2
      return
 1000 continue
c
c    **         Order  One             **
c
      r13 = r1**3.d0
      r23 = r2**3.d0
      go to (1001, 1002, 1003, 1004), i

 1001 h = -x(4) + xmua*qa/r13 + xmu*qb/r23
      return
 1002 h = x(2) + x(3)
      return
 1003 h = x(2) + xmua*x(3)/r13 + xmu*x(3)/r23
      return
 1004 h = x(4) - x(1)
      return

 2000 continue
c
c    **         Order  Two             **
c
      r13 = r1**3.d0
      r23 = r2**3.d0
      r15 = r1**5.d0
      r25 = r2**5.d0

      go to (2001, 2002, 2003, 2004),i
 2001 go to (2011, 2012, 2013, 2014),j
 2002 go to (2021, 2022, 2023, 2024),j
 2003 go to (2031, 2032, 2033, 2034),j
 2004 go to (2041, 2042, 2043, 2044),j

 2011 h = xmua/r13 + xmu/r23 -3.d0*xmua*qa*qa/r15
     1     -3.d0*xmu*qb*qb/r25
      return
 2012 h = 0.d0
      return
 2013 h = -3.d0*xmua*qa*x(3)/r15 - 3.d0*xmu*qb*x(3)/r25
      return
 2014 h = -1.d0
      return
 2021 h = 0.d0
      return
 2022 h = 1.d0
      return
 2023 h = 1.d0
      return
 2024 h = 0.d0
      return
 2031 go to 2013
```

```
 2032 h = 1.d0
      return
 2033 h = -3.d0*xmua*x(3)*x(3)/r15 - 3.d0*xmu*x(3)*x(3)/r25
     1     + xmua/r13 + xmu/r23
      return
 2034 h = 0.d0
      return
 2041 h = -1.d0
      return
 2042 h = 0.d0
      return
 2043 h = 0.d0
      return
 2044 h = 1.d0
      return

 3000 continue
c
c    **         Order  Three           **
c
      r15 = r1**5.d0
      r25 = r2**5.d0
      r17 = r1**7.d0
      r27 = r2**7.d0

      go to (30001, 30002, 30003, 30004),i
30001 go to (30110, 30120, 30130, 30140),j
30002 go to (30210, 30220, 30230, 30240),j
30003 go to (30310, 30320, 30330, 30340),j
30004 go to (30410, 30420, 30430, 30440),j
c   note matrix is quite sparse now.......
30110 go to (30111, 30112, 30113, 30114),k
30130 go to (30131, 30132, 30133, 30134),k
30310 go to (30311, 30312, 30313, 30314),k
30330 go to (30331, 30332, 30333, 30334),k

30111 h = -9.d0*xmua*qa/r15 - 9.d0*xmu*qb/r25
     1  + 15.d0*xmua*qa*qa*qa/r17 + 15.d0*xmu*qb*qb*qb/r27
      return
30112 h = 0.d0
      return
30113 h = -3.d0*xmua*x(3)/r15 - 3.d0*xmu*x(3)/r25
     1   + 15.d0*xmua*qa*qa*x(3)/r17 + 15.d0*xmu*qb*qb*x(3)/r27
      return
30114 h = 0.d0
      return
30120 h = 0.d0
      return
30131 go to 30113
30132 h = 0.d0
      return
30133 h = -3.d0*xmua*qa/r15 - 3.d0*xmu*qb/r25
     1     + 15.d0*xmua*qa*x(3)*x(3)/r17 + 15.d0*xmu*qb*x(3)*x(3)/r27
      return
30134 h = 0.d0
      return
30140 h = 0.d0
      return
30210 h = 0.d0
      return
30220 h = 0.d0
      return
30230 h = 0.d0
      return
30240 h = 0.d0
      return
30311 go to 30113
30312 h = 0.d0
      return
30313 go to 30133
30314 h = 0.d0
      return
30320 h = 0.d0
      return
30331 go to 30133
30332 h = 0.d0
      return
30333 h = -9.d0*xmua*x(3)/r15 - 9.d0*xmu*x(3)/r25
     1   + 15.d0*(xmua/r17 + xmu/r27)*x(3)*x(3)*x(3)
      return
30334 h = 0.d0
      return
30340 h = 0.d0
      return
30410 h = 0.d0
      return
```

```
30420 h = 0.d0
      return
30430 h = 0.d0
      return
30440 h = 0.d0

      return
      end
```

```
ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc
c     SUBROUTINE FOURIER
c
c     PURPOSE: harmonic analysis of 2n values of function F
c              evenly spaced at interval 2pi/2n, starting with
c              zero, into n+1 cosine coefficients ck and n-1 sine
c              coefficients sk.
c
c     ref Brouwer and Clemence, p 109
c
ccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccccc

      subroutine fourier(F,ck,sk,n)

      double precision F(2),ck(2),sk(2),twopi,alpha

      twopi = 2.d0*3.141592653589d0
      alpha = twopi/dble(2*n)
      n2m1 = 2*n-1
c
c     order k loop
c
      do 500 k = 0,n
c
c          cosine sum
c
           ck(k+1) = 0.d0

           do 200 j = 0,n2m1
              ck(k+1) = ck(k+1) + F(j+1) * dcos( dble(k*j)*alpha)
  200      continue

           ck(k+1) = ck(k+i)/dble(n)
c
c          sine sum
c
           if(k .eq. 0) go to 500
           if(k .eq. n) go to 500
           sk(k+1) = 0.d0
           do 400 j = 1,n2m1
              sk(k+1) = sk(k+1) + F(j+1) * dsin( dble(k*j)*alpha)
  400      continue
           sk(k+1) = sk(k+1)/dble(n)

  500 continue
c
c     correct first and last cosine coefficient
c
      ck(1) = 0.5d0*ck(1)
      ck(n+1) = 0.5d0*ck(n+1)

      return
      end
```

Bibliography

1. Brouwer, Dirk and Gerald M. Clemence. Methods of Celestial Mechanics, New York and London: Academic Press, 1961.

2. Calico, Robert A. and William E. Wiesel "Control of Time-Periodic Systems,"Journal of Guidance, Control, and Dynamics, 7: 671-676 (November-December 1984).

3. Jefferys, William H. An Atlas of Surface of Section For the Restricted Problem of Three Bodies, University of Texas at Austin: Applied Mechanics Research Laboratory, 1971.

4. Pars, L.A. A Treatise on Analytical Dynamics, New York, N.Y.: John Wiley and Sons, 1965.

5. Szebehely, Victor Theory of Orbits, (The Restricted Problem of Three Bodies), New York and London: Academic Press, 1967.

6. Waltman, Paul A Second Course in Elementary Differential Equations, Orlando: Academic Press, 1986.

7. Wiesel, William E. "Perturbation Theory in the Vicinity of a Periodic Orbit by Repeated Linear Transformations," Journal of Celestial Mechanics, 23: 231-242 (1981).

8. Wiesel, William E. Spaceflight Dynamics, New York: McGraw-Hill Book Company, 1989.

9. Wiesel, William E., Professor of Astronautics, Course text distributed in Mech 636, Advanced Astrodynamics. School of Engineering, Air Force Institute of Technology (AU), Wright-Patterson AFB OH, January 1991.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY (Leave blank)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	Dec 1991	Master's Thesis

4. TITLE AND SUBTITLE

Perturbation Theory for Restricted Three-Body Orbits

5. FUNDING NUMBERS

6. AUTHOR(S)

David A. Ross, Captain, USAF

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Air Force Institute of Technology, WPAFB OH 45433-6583

8. PERFORMING ORGANIZATION REPORT NUMBER

AFIT/GA/ENY/91D-7

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT

Unlimited

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

A perturbation theory for restricted three-body orbits, using a periodic trajectory as the reference solution, is investigated. Nearly-periodic orbits are integrated and then twice transformed into a new set of variables. Plots of these orbits are presented as an exact representation of nearly-periodic motion. The hamiltonian is then canonically transformed and expanded in a Taylor's series. The expansion is truncated after the fourth term. An approximation for the equations of motion for nearly-periodic orbits is then derived and tested. The expanded trajectories are then compared to the exact case. The expanded equations work well in limited region surrounding the periodic orbit. It is postulated, however, that the expanded equations could be improved by retaining more terms in the hamiltonian expansion.

14. SUBJECT TERMS

Restricted Three-body Perturbation Theory
Periodic Orbit Reference Solution

15. NUMBER OF PAGES
103

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	Unlimited

GENERAL INSTRUCTIONS FOR COMPLETING SF 298

The Report Documentation Page (RDP) is used in announcing and cataloging reports. It is important that this information be consistent with the rest of the report, particularly the cover and title page. Instructions for filling in each block of the form follow. It is important to ***stay within the lines*** to meet ***optical scanning requirements***.

Block 1. Agency Use Only *(Leave blank)*.

Block 2. Report Date. Full publication date including day, month, and year, if available (e.g. 1 Jan 88). Must cite at least the year.

Block 3. Type of Report and Dates Covered. State whether report is interim, final, etc. If applicable, enter inclusive report dates (e.g. 10 Jun 87 - 30 Jun 88).

Block 4. Title and Subtitle. A title is taken from the part of the report that provides the most meaningful and complete information. When a report is prepared in more than one volume, repeat the primary title, add volume number, and include subtitle for the specific volume. On classified documents enter the title classification in parentheses.

Block 5. Funding Numbers. To include contract and grant numbers; may include program element number(s), project number(s), task number(s), and work unit number(s). Use the following labels:

C	- Contract	**PR**	- Project
G	- Grant	**TA**	- Task
PE	- Program Element	**WU**	- Work Unit Accession No.

Block 6. Author(s). Name(s) of person(s) responsible for writing the report, performing the research, or credited with the content of the report. If editor or compiler, this should follow the name(s).

Block 7. Performing Organization Name(s) and Address(es). Self-explanatory.

Block 8. Performing Organization Report Number. Enter the unique alphanumeric report number(s) assigned by the organization performing the report.

Block 9. Sponsoring/Monitoring Agency Name(s) and Address(es). Self-explanatory.

Block 10. Sponsoring/Monitoring Agency Report Number. *(If known)*

Block 11. Supplementary Notes. Enter information not included elsewhere such as: Prepared in cooperation with...; Trans. of...; To be published in.... When a report is revised, include a statement whether the new report supersedes or supplements the older report.

Block 12a. Distribution/Availability Statement. Denotes public availability or limitations. Cite any availability to the public. Enter additional limitations or special markings in all capitals (e.g. NOFORN, REL, ITAR).

- **DOD** - See DoDD 5230.24, "Distribution Statements on Technical Documents."
- **DOE** - See authorities.
- **NASA** - See Handbook NHB 2200.2.
- **NTIS** - Leave blank.

Block 12b. Distribution Code.

- **DOD** - Leave blank.
- **DOE** - Enter DOE distribution categories from the Standard Distribution for Unclassified Scientific and Technical Reports.
- **NASA** - Leave blank.
- **NTIS** - Leave blank.

Block 13. Abstract. Include a brief *(Maximum 200 words)* factual summary of the most significant information contained in the report.

Block 14. Subject Terms. Keywords or phrases identifying major subjects in the report.

Block 15. Number of Pages. Enter the total number of pages.

Block 16. Price Code. Enter appropriate price code *(NTIS only)*.

Blocks 17. - 19. Security Classifications. Self-explanatory. Enter U.S. Security Classification in accordance with U.S. Security Regulations (i.e., UNCLASSIFIED). If form contains classified information, stamp classification on the top and bottom of the page.

Block 20. Limitation of Abstract. This block must be completed to assign a limitation to the abstract. Enter either UL (unlimited) or SAR (same as report). An entry in this block is necessary if the abstract is to be limited. If blank, the abstract is assumed to be unlimited.

*U.S.GPO 1989-0-273-271